Union internationale des télécommunications

UIT-T

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

M.3020

(07/2011)

SÉRIE M: GESTION DES TÉLÉCOMMUNICATIONS Y COMPRIS LE RGT ET MAINTENANCE DES RÉSEAUX

Réseau de gestion des télécommunications

Méthodologie pour la spécification des interfaces de gestion

Recommandation UIT-T M.3020

RECOMMANDATIONS UIT-T DE LA SÉRIE M

GESTION DES TÉLÉCOMMUNICATIONS Y COMPRIS LE RGT ET MAINTENANCE DES RÉSEAUX

Introduction et principes généraux de maintenance et organisation de la maintenance	M.10–M.299
Systèmes de transmission internationaux	M.300–M.559
Circuits téléphoniques internationaux	M.560–M.759
Systèmes de signalisation à canal sémaphore	M.760–M.799
Systèmes internationaux de télégraphie et de phototélégraphie	M.800–M.899
Liaisons internationales louées par groupes primaires et secondaires	M.900–M.999
Circuits internationaux loués	M.1000–M.1099
Systèmes et services de télécommunication mobile	M.1100–M.1199
Réseau téléphonique public international	M.1200–M.1299
Systèmes internationaux de transmission de données	M.1300–M.1399
Appellations et échange d'informations	M.1400–M.1999
Réseau de transport international	M.2000–M.2999
Réseau de gestion des télécommunications	**M.3000–M.3599**
Réseaux numériques à intégration de services	M.3600–M.3999
Systèmes de signalisation par canal sémaphore	M.4000–M.4999

Pour plus de détails, voir la Liste des Recommandations de l'UIT-T.

Recommandation UIT-T M.3020

Méthodologie pour la spécification des interfaces de gestion

Résumé

La Recommandation UIT-T M.3020 présente la méthodologie pour la spécification des interfaces de gestion (MSIG). Elle décrit le processus permettant de spécifier des interfaces à partir des besoins, de l'analyse et de la conception (RAD, *requirements, analysis and design*). Des lignes directrices sont données en vue de décrire les phases RAD au moyen du langage de modélisation unifié (UML, *unified modelling language*); toutefois, elles n'excluent pas d'autres techniques de spécification d'interface. Dans la présente Recommandation UIT-T, les lignes directrices relatives à l'utilisation du langage UML sont décrites à un haut niveau.

Historique

Edition	Recommandation	Approbation	Commission d'études
1.0	ITU-T M.3020	1992-10-05	
2.0	ITU-T M.3020	1995-07-27	4
3.0	ITU-T M.3020	2000-02-04	4
4.0	ITU-T M.3020	2007-07-22	4
5.0	ITU-T M.3020	2008-07-29	4
6.0	ITU-T M.3020	2009-05-14	2
7.0	ITU-T M.3020	2010-09-06	2
8.0	ITU-T M.3020	2011-07-14	2

AVANT-PROPOS

L'Union internationale des télécommunications (UIT) est une institution spécialisée des Nations Unies dans le domaine des télécommunications et des technologies de l'information et de la communication (ICT). Le Secteur de la normalisation des télécommunications (UIT-T) est un organe permanent de l'UIT. Il est chargé de l'étude des questions techniques, d'exploitation et de tarification, et émet à ce sujet des Recommandations en vue de la normalisation des télécommunications à l'échelle mondiale.

L'Assemblée mondiale de normalisation des télécommunications (AMNT), qui se réunit tous les quatre ans, détermine les thèmes d'étude à traiter par les Commissions d'études de l'UIT-T, lesquelles élaborent en retour des Recommandations sur ces thèmes.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution 1 de l'AMNT.

Dans certains secteurs des technologies de l'information qui correspondent à la sphère de compétence de l'UIT-T, les normes nécessaires se préparent en collaboration avec l'ISO et la CEI.

NOTE

Dans la présente Recommandation, l'expression "Administration" est utilisée pour désigner de façon abrégée aussi bien une administration de télécommunications qu'une exploitation reconnue.

Le respect de cette Recommandation se fait à titre volontaire. Cependant, il se peut que la Recommandation contienne certaines dispositions obligatoires (pour assurer, par exemple, l'interopérabilité et l'applicabilité) et considère que la Recommandation est respectée lorsque toutes ces dispositions sont observées. Le futur d'obligation et les autres moyens d'expression de l'obligation comme le verbe "devoir" ainsi que leurs formes négatives servent à énoncer des prescriptions. L'utilisation de ces formes ne signifie pas qu'il est obligatoire de respecter la Recommandation.

DROITS DE PROPRIÉTÉ INTELLECTUELLE

L'UIT attire l'attention sur la possibilité que l'application ou la mise en œuvre de la présente Recommandation puisse donner lieu à l'utilisation d'un droit de propriété intellectuelle. L'UIT ne prend pas position en ce qui concerne l'existence, la validité ou l'applicabilité des droits de propriété intellectuelle, qu'ils soient revendiqués par un membre de l'UIT ou par une tierce partie étrangère à la procédure d'élaboration des Recommandations.

A la date d'approbation de la présente Recommandation, l'UIT n'avait pas été avisée de l'existence d'une propriété intellectuelle protégée par des brevets à acquérir pour mettre en œuvre la présente Recommandation. Toutefois, comme il ne s'agit peut-être pas de renseignements les plus récents, il est vivement recommandé aux développeurs de consulter la base de données des brevets du TSB sous http://www.itu.int/ITU-T/ipr/.

TABLE DES MATIÈRES

Page

Recommandation UIT-T M.3020

Méthodologie pour la spécification des interfaces de gestion

1 Domaine d'application

La présente Recommandation décrit la méthodologie pour la spécification des interfaces de gestion (MSIG). Elle décrit le processus permettant de spécifier des interfaces machine-machine à partir des besoins, de l'analyse et de la conception (RAD, *requirements, analysis and design*). Des lignes directrices sont données en vue de décrire les phases RAD au moyen du langage de modélisation unifié (UML, *unified modelling language*); toutefois, elles n'excluent pas d'autres techniques de spécification d'interface. Dans la présente Recommandation, les lignes directrices relatives à l'utilisation du langage UML sont décrites à un haut niveau. Une spécification d'interface s'applique au(x) service(s) de gestion défini(s) dans [UIT-T M.3200] et/ou vient à l'appui des processus de gestion définis dans la série [UIT-T M.3050.x]. Une telle spécification peut prendre en charge tout ou partie d'un service de gestion ou plusieurs services de gestion. Un service de gestion est composé de fonctions de gestion. Ces fonctions peuvent renvoyer à celles définies dans [UIT-T M.3400] ou aux processus définis dans la série [UIT-T M.3050.x] ou être spécifiques à un domaine géré particulier. En outre, de nouvelles fonctions peuvent être identifiées au besoin.

La méthodologie s'applique aussi bien au type traditionnel gestionnaire/agent des interfaces de gestion [UIT-T M.3010] qu'aux principes de l'architecture orientée vers les services (SOA, *service oriented architecture*) adoptés pour l'architecture de gestion des réseaux de prochaine génération [UIT-T M.3060].

2 Références

La présente Recommandation se réfère à certaines dispositions des Recommandations UIT-T et textes suivants qui, de ce fait, en sont partie intégrante. Les versions indiquées étaient en vigueur au moment de la publication de la présente Recommandation. Toute Recommandation ou tout texte étant sujet à révision, les utilisateurs de la présente Recommandation sont invités à se reporter, si possible, aux versions les plus récentes des références normatives suivantes. La liste des Recommandations de l'UIT-T en vigueur est régulièrement publiée. La référence à un document figurant dans la présente Recommandation ne donne pas à ce document, en tant que tel, le statut d'une Recommandation.

[UIT-T M.3010] Recommandation UIT-T M.3010 (2000), *Principes des réseaux de gestion des télécommunications.*

[UIT-T M.3050.x] Recommandation UIT-T M.3050.x (2007), *Plan amélioré d'exploitation des télécommunications (eTOM).*

[UIT-T M.3060] Recommandation UIT-T M.3060/Y.2401 (2006), *Principes pour la gestion des réseaux de prochaine génération.*

[UIT-T M.3200] Recommandation UIT-T M.3200 (1997), *Services de gestion du réseau de gestion des télécommunications et domaines gérés des télécommunications: aperçu général.*

[UIT-T M.3400] Recommandation UIT-T M.3400 (2000), *Fonctions de gestion du réseau de gestion des télécommunications.*

[UIT-T Q.812] Recommandation UIT-T Q.812 (2004), *Profils des protocoles des couches supérieures pour les interfaces Q et X.*

[UIT-T X.680] Recommandation UIT-T X.680 (2008) | ISO/CEI 8824-1:2008, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification de la notation de base.*

[UIT-T X.681] Recommandation UIT-T X.681 (2008) | ISO/CEI 8824-2:2008, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification des objets informationnels.*

[UIT-T X.722] Recommandation UIT-T X.722 (1992) | ISO/CEI 10165-4:1992, *Technologies de l'information – Interconnexion des systèmes ouverts – Structure des informations de gestion: directives pour la définition des objets gérés.*

[UIT-T Z.100] Recommandation UIT-T Z.100 (2007), *Langage de description et de spécification – Présentation générale de SDL-2010*.

[OMG UML] OMG: *Spécification du langage de modélisation unifié UML, version 1.5*.

On trouvera dans la bibliographie une liste de documents de référence non normatifs.

3 Définitions

3.1 Termes définis ailleurs

La présente Recommandation utilise les termes suivants tirés de [UIT-T M.3010]:

– utilisateur;
– service de gestion;
– ensemble de fonctions de gestion.

La présente Recommandation utilise les termes suivants tirés de [OMG UML]:

– diagramme d'activité;
– acteur;
– association;
– classe;
– diagramme de classes;
– classificateur;
– diagramme de collaboration;
– composition;
– modelElement;
– diagramme de séquence;
– diagramme d'états;
– stéréotype;
– cas d'utilisation.

La présente Recommandation utilise le terme suivant tiré de [UIT-T M.3060]:

– point de référence.

3.2 Termes définis dans la présente Recommandation

La présente Recommandation définit les termes suivants:

3.2.1 agent: Encapsule un sous-ensemble bien défini de fonctions de gestion. Il communique avec les gestionnaires au moyen d'une interface de gestion. Pour le gestionnaire, le comportement de l'agent n'est visible que par l'intermédiaire de l'interface de gestion.

NOTE – Considéré comme équivalent à IRPAgent (agent de point de référence d'intégration) [b-3GPP TS 32.150].

3.2.2 classe d'objets informationnels (IOC, *information object class*): Décrit les informations qui peuvent être acheminées/utilisées dans les interfaces de gestion; l'IOC est modélisée à l'aide de la "classe" stéréotype du métamodèle UML. Pour une définition formelle de la classe d'objets informationnels et de sa structure de spécification, voir l'Annexe B.

3.2.3 service d'information: Décrit les informations relatives aux entités à gérer (ressources de réseau ou objets supports) ainsi que la façon dont ces informations peuvent être gérées pour un domaine fonctionnel donné. Des services d'information sont définis pour tous les points IRP (points de référence d'intégration, *Integration Reference Points*).

NOTE – Considéré comme identique à la définition du service d'information figurant dans [b-3GPP TS 32.150].

3.2.4 type d'information: Spécification du type des paramètres d'entrée des opérations.

3.2.5 point de référence d'intégration: Concept architectural décrit par un ensemble de spécifications aux fins de la définition d'un certain aspect de l'interface de gestion, comprenant une spécification des besoins, une spécification du service d'information et une ou plusieurs spécifications d'ensembles de solutions.

NOTE – Considéré comme identique à la définition du point IRP figurant dans [b-3GPP TS 32.150].

3.2.6 objectifs de gestion: Objectifs de haut niveau d'un utilisateur exécutant des activités de gestion.

3.2.7 interface de gestion: Réalisation des capacités de gestion entre un gestionnaire et un agent permettant à un seul et même gestionnaire d'utiliser plusieurs agents et à un agent de venir à l'appui de plusieurs gestionnaires.

NOTE – Q, C2B/B2B et Itf-N (3GPP) sont des exemples d'interface de gestion.

3.2.8 rôle de gestion: Définit les activités attendues du personnel ou des systèmes d'exploitation qui réalisent la gestion des télécommunications. Ces rôles sont définis indépendamment des autres composants, c'est-à-dire des ressources de télécommunication et des fonctions de gestion.

3.2.9 scénario de gestion: Exemple d'interactions de gestion de la part d'un service de gestion.

3.2.10 gestionnaire: Modélise l'utilisateur d'un ou plusieurs agents et interagit directement avec le ou les agents au moyen d'interfaces de gestion.

Etant donné que le gestionnaire représente un utilisateur d'agent, il donne une image précise de ce que l'agent est censé accomplir. Pour l'agent, le comportement du gestionnaire n'est visible que par l'intermédiaire de l'interface de gestion.

NOTE – Considéré comme équivalent à IRPManager (gestionnaire de point de référence d'intégration) [b-3GPP TS 32.150].

3.2.11 information de correspondance (*matching information*): Spécification du type d'un paramètre (éventuellement référence à la classe IOC ou à un attribut de la classe IOC).

3.2.12 spécification indépendante du protocole: Définit les interfaces de gestion qui viennent à l'appui des capacités de gestion sans qu'il y ait à se préoccuper du protocole et de la représentation des informations implicites ou requis par CORBA, XML, etc.

3.2.13 spécification dépendante du protocole: Définit les interfaces de gestion qui viennent à l'appui des capacités de gestion pour un choix particulier de technologie de gestion (CORBA par exemple).

NOTE – Considéré comme équivalent à l'ensemble des solutions [b-3GPP TS 32.150].

3.2.14 ressources de télécommunication: entités physiques ou logiques nécessitant une gestion par l'intermédiaire de services de gestion.

4 Abréviations

La présente Recommandation utilise les abréviations suivantes:

3GPP projet de partenariat de 3ème génération (*third generation partnership project*)

ADM administratif (contexte: catégorie de besoins)

ASN.1 notation de syntaxe abstraite numéro un (*abstract syntax notation one*)

CM conditionnel-obligatoire (*conditional-mandatory*)

CO conditionnel-optionnel (*conditional-optional*)

CON conceptuel (contexte: catégorie de besoins)

CORBA architecture de courtier commun de requête d'objets (*common object request broker architecture*)

ES ensemble de solutions

FON fonctionnel (contexte: catégorie de besoins)

GDMO directives pour la définition des objets gérés (*guidelines for the definition of managed objects*).

IDL langage de définition d'interface (*interface definition language*)

IOC classe d'objets informationnels (*information object class*)

IRP point de référence d'intégration (*integration reference point*)

IS service d'information (*information service*)

MSIG méthodologie pour la spécification des interfaces de gestion

NA non applicable

NE élément de réseau (*network element*)

NON non fonctionnel (contexte: catégorie de besoins)

OMG Object Management Group

OO orienté objet

OSI interconnexion des systèmes ouverts (*open systems interconnection*)

SDL langage de description et de spécification (*specification and description language*)

SOA architecture orientée vers les services (*service-oriented architecture*)

ST spécifications techniques

UML langage de modélisation unifié (*unified modelling language*)

XML langage de balisage extensible (*extensible markup language*)

5 Conventions

Le § A.1 contient les conventions applicables à la phase d'expression des besoins.

Le § B.1 contient les conventions applicables à la phase d'analyse.

6 Fondements de la méthodologie et de la notation

Les critères utilisés pour élaborer la méthodologie et choisir la notation sont les suivants:

1) La méthodologie, y compris le choix de la notation, doit prendre en compte tous les besoins de l'espace du problème, à savoir la gestion des télécommunications.

2) La méthodologie facilite l'expression des besoins, l'analyse (services d'information) correspondante et l'élaboration des spécifications de conception (ensembles de solutions) correspondantes.

3) La notation doit faciliter l'expression non ambiguë de la spécification dans le profil de protocole de gestion cible. La méthodologie ne traite pas des choix possibles de services de protocole (service de sécurité de CORBA par exemple).

NOTE – Les protocoles de gestion applicables aux normes de l'UIT-T figurent dans [UIT-T Q.812].

4) La méthodologie doit permettre de spécifier les éléments obligatoires et les éléments optionnels dans les trois phases. Elle spécifie aussi la façon de relier les éléments obligatoires ou optionnels entre les trois phases.

5) Il doit être possible de produire, à partir de la spécification indépendante du protocole (analyse|IS), des définitions interopérables propres au langage utilisé, à savoir Conception|ES (par exemple UML avec IDL, UML avec GDMO/ASN.1).

7 Méthodologie

7.1 Généralités

L'objet de la présente méthodologie est de décrire les processus conduisant à la définition des interfaces de gestion machine-machine.

7.2 Application et structure de la méthodologie

La méthodologie pour la spécification des interfaces de gestion (MSIG) spécifie un processus à trois phases. Les caractéristiques de ce processus permettent la traçabilité à travers ces trois phases. Les trois phases appliquent des techniques admises par l'industrie qui reposent sur des principes d'analyse et de conception orientés objet. Ces trois phases sont l'expression des besoins, l'analyse et la conception. Les techniques en question doivent permettre l'utilisation ou le développement d'outils d'aide disponibles sur le marché. Différentes techniques peuvent être utilisées selon les phases, en fonction de la nature du problème.

7.3 Méthodologie détaillée

7.3.1 Généralités

Les phases d'expression des besoins et d'analyse produisent des spécifications en langage UML. La phase de conception utilise la notation spécifique au paradigme de gestion de réseau. Les résultats des trois phases sont les suivants:

– Phase d'expression des besoins – Besoins.

– Phase d'analyse – Spécification indépendante de l'implémentation.

– Phase de conception – Spécification dépendante de la technologie.

Pour la phase de conception, on utilisera, dans un premier temps, une approche manuelle ou personnalisée. Lorsqu'une définition interopérable dépendante du protocole peut être générée au moyen d'outils, la notation UML pourra être appliquée à cette phase.

Les paragraphes ci-dessous décrivent les trois phases.

7.3.2 Phase d'expression des besoins

Les besoins correspondant au problème à résoudre se répartissent en deux classes. La première classe de besoins est appelée ici "besoins de niveau métier". Un spécialiste du domaine doit être en mesure de déterminer si les besoins spécifiés couvrent bien les besoins du problème de gestion à résoudre. La seconde classe est appelée "besoins de niveau spécification". Ces besoins doivent être suffisamment détaillés pour permettre de définir les interfaces pendant les phases d'analyse et de conception. Etant donné que les définitions finales des interfaces doivent être traçables vis-à-vis des besoins, il peut être nécessaire de prévoir des allers-retours entre les trois phases. Toute ambiguïté dans les besoins devra être levée par ce processus d'allers-retours afin de garantir que la spécification peut être développée et implémentée.

Des données d'interface homme-machine peuvent être spécifiées dans la seconde classe des besoins. Ces besoins peuvent avoir des répercussions considérables sur les concepts et les données définis dans les phases suivantes. Pour plus de détails, voir l'Appendice III et les Recommandations de la série UIT-T M.1400 sur la conception des données pour les interfaces homme-machine.

Différentes techniques peuvent être utilisées pour spécifier les deux classes de besoins susmentionnés. Quelle que soit la technique utilisée, la lisibilité des besoins revêt un caractère essentiel. Il n'est pas exigé que ceux-ci soient exprimés en notation lisible par machine dès l'instant où la lisibilité et la traçabilité sont assurées. La solution recommandée pour cerner les différents besoins en assurant leur traçabilité est de les énumérer.

La phase d'expression des besoins couvre plusieurs aspects: politique de sécurité, étendue du domaine du problème en termes d'applications, ressources, rôle des ressources, etc. Les besoins définissent, pour l'espace du problème, les rôles, les responsabilités et les relations entre les entités constitutives. Différentes techniques, notamment la représentation textuelle, peuvent être utilisées pour spécifier les besoins métier. Pour que ces besoins soient facilement traçables vis-à-vis des phases de conception et d'implémentation, il est recommandé de les énumérer.

Le problème doit être délimité, ce qui suppose de définir précisément son domaine d'application. Une façon de procéder consiste à utiliser les services de gestion recensés dans [UIT-T M.3200] ainsi que les ensembles de fonctions définis dans [UIT-T M.3400]. Les besoins sont exprimés au moyen des ressources à gérer et des fonctions de gestion. Autre solution: suivre la démarche reposant sur les processus métier intitulée "Plan amélioré d'exploitation des télécommunications (eTOM)" et décrite dans [UIT-T M.3050.x].

La relation entre les démarches proposées dans [UIT-T M.3200] et [UIT-T M.3050] est décrite dans [UIT-T M.3050.x].

Les fonctions de gestion doivent être regroupées et prises en charge au sein d'applications traitant de besoins métier spécifiques. Pour faciliter le regroupement clair et efficace de ces fonctions, il est important de relier les processus eTOM, les services de gestion énumérés dans [UIT-T M.3200] et les fonctions de gestion et ensembles de fonctions de gestion énumérés dans [UIT-T M.3400]. Il pourra être nécessaire d'étendre [UIT-T M.3400] pour répondre aux besoins métier du problème.

Il convient d'utiliser des cas d'utilisation UML et des scénarios pour dialoguer avec les spécialistes du domaine lors de la consignation des besoins de niveau métier. Les besoins doivent également mettre en évidence les états de dysfonctionnement visibles au niveau du processus métier.

NOTE – Il n'est pas exigé que tous les besoins soient exprimés sous forme de cas d'utilisation.

Les besoins exprimés doivent être complets et détaillés. La nature récursive de la méthodologie permet d'obtenir cette exhaustivité. Les phases d'analyse et de conception dépendent de la bonne formulation des besoins (clairs et bien documentés).

Des lignes directrices et un modèle pour la structure et le recensement des besoins sont décrits au § A.1.2.

Les cas d'utilisation sont des objectifs, qui sont réalisés par une séquence d'étapes, chaque étape pouvant être vue comme un sous-objectif du cas d'utilisation. Chaque étape représente soit un autre cas d'utilisation (cas d'utilisation de niveau inférieur), soit une action autonome, qui constitue le plus bas niveau de décomposition du cas.

Des lignes directrices et un modèle pour l'élaboration des cas d'utilisation sont présentés au § A.1.2.

L'Appendice I fournit un exemple de définition des besoins.

7.3.3 Analyse

La phase d'analyse consiste à identifier, à partir des besoins, les différentes entités qui interagissent, leurs propriétés et leurs interrelations, et, partant, à définir les interfaces offertes par ces entités. Dans la notation UML, ces entités deviennent des classes. Les descriptions des classes ainsi que les interfaces exposées doivent être traçables par rapport aux besoins. A noter que la relation entre les classes définies dans la phase d'analyse et les classes définies dans la phase de conception n'est pas nécessairement biunivoque.

La phase d'analyse doit prendre en compte les besoins en termes de données d'interface homme-machine (autrement dit, le modèle d'information doit être suffisamment riche pour que l'on puisse effectuer la conception à partir des résultats de l'analyse).

La présente Recommandation contient des directives de haut niveau sur la façon d'utiliser la notation UML pour spécifier les interfaces de gestion; cela étant, il est possible d'utiliser le langage SDL [UIT-T Z.100] pour compléter les définitions UML.

La phase d'analyse doit être indépendante des contraintes de conception. Par exemple, l'analyse peut être documentée en utilisant des principes orientés objet même si la conception repose sur une technologie non orientée objet. Les informations spécifiées dans la phase d'analyse comprennent les descriptions de classe, les définitions de données, les relations de classe, les diagrammes d'interaction (diagrammes de séquence et/ou diagrammes de collaboration), les diagrammes états-transitions et les diagrammes d'activité. La définition d'une classe comprend la spécification des opérations, des notifications, des attributs et du comportement, ces éléments étant consignés sous forme de notes ou de description textuelle.

Dans un souci d'harmonisation des interfaces de gestion, il convient de réutiliser, pendant la phase d'analyse, les services de gestion communs indépendants du protocole (le cas échéant) ou d'autres services existants.

Des lignes directrices et un modèle pour l'élaboration des cas d'utilisation sont présentés à l'Annexe A.

Le modèle d'analyse utilise le type d'information comme l'une des caractéristiques pour décrire les attributs des classes d'objets informationnels (IOC) ainsi que les paramètres d'opération/de notification. Le ou les types d'information valides pouvant être utilisés sont recensés à l'Annexe E avec leurs significations respectives.

7.3.4 Conception

7.3.4.1 Généralités

La phase de conception consiste à produire une spécification d'interfaces interopérables et pouvant être implémentées. Cette activité comprend le choix d'un langage de spécification cible. Les spécifications de la phase de conception dépendent du paradigme de gestion (IDL pour les interfaces CORBA par exemple).

Dans cette phase, on distingue trois types de spécifications de données: conception des données à transmettre par l'intermédiaire de multiples interfaces (panne et qualité de fonctionnement par exemple), qui dépend du paradigme de gestion (XML, etc.); messages (rapports d'alarme, etc.) à

transmettre sur chaque interface; méthode de codage des données (XML compressé par exemple) compatible avec un paradigme en particulier.

Le choix du paradigme de gestion est traité dans d'autres Recommandations UIT-T. Un aperçu général en est donné dans les paragraphes suivants.

Dans la phase de conception, il est recommandé de faire des renvois aux descriptions UML des phases d'expression des besoins et d'analyse pour étoffer la spécification du comportement. Par exemple, la définition des aspects comportementaux selon les directives GDMO peut renvoyer à des diagrammes d'états, des diagrammes de séquence et des définitions de classe issus de la phase d'analyse. Au besoin, on peut intégrer des diagrammes UML supplémentaires décrivant les interactions entre entités, qui correspondent à des paradigmes dépendants du protocole.

A mesure que d'autres paradigmes sont adoptés pour gérer les réseaux, les notations/langages définis par ces paradigmes sont utilisés.

7.3.4.2 CORBA

Dans le contexte de la gestion fondée sur CORBA, le modèle d'information est défini au moyen du langage IDL.

7.3.4.3 GDMO

Dans le contexte du paradigme reposant sur la gestion des systèmes OSI [UIT-T X.722], la spécification de conception correspond à la spécification du modèle d'information, qui est élaborée au moyen des modèles GDMO concernant les classes d'objets gérés, les attributs, le comportement, les notifications, les actions, les instances de dénomination de la classe et les spécifications d'erreur/exception. La syntaxe de l'information est spécifiée au moyen de la notation ASN.1 [UIT-T X.680].

En GDMO, la hiérarchie des classes d'objets spécifie les propriétés des classes qui sont nécessaires à la gestion. Il est nécessaire d'utiliser largement l'héritage (hyperclasses et sous-classes) afin de tirer le meilleur parti de la réutilisation des spécifications. Les classes d'objets sont spécifiées à l'aide des modèles tirés de [UIT-T X.722]. Les modèles définissant le modèle d'information doivent être enregistrés (conformément aux règles définies dans [UIT-T X.722]) en attribuant une valeur à l'identificateur d'objet ASN.1. S'agissant des classes d'objets qui sont déjà spécifiées dans d'autres Recommandations UIT-T et normes ISO, seul un renvoi à la Recommandation et à la classe d'objets en question est nécessaire. La dénomination ne fait pas partie de la hiérarchie des classes d'objets; ce n'est pas non plus sa finalité.

7.3.4.4 XML

Ce point fera l'objet d'une étude complémentaire.

8 Spécifications des interfaces de gestion

Une spécification d'interfaces de gestion comprend les spécifications décrites au § 7, à savoir l'expression des besoins, l'analyse et la conception. Les Annexes A, B et C fournissent une structure d'élaboration de ces spécifications.

Même si la conception des systèmes n'est pas considérée comme faisant partie des Recommandations UIT T relatives à la gestion, ces techniques et les notations qui les accompagnent sont également applicables aux spécifications des interfaces de gestion lors de la conception d'un système. Elles permettent de décrire plus aisément comment les spécifications d'interfaces peuvent gérer les ressources d'un système, par exemple un élément de réseau.

9 Traçabilité dans le processus MSIG

Pour assurer la traçabilité des besoins, de l'analyse et de la conception, il est nécessaire de procéder à une identification appropriée. La traçabilité est assurée par des renvois entre les entités spécifiées à l'intérieur de chaque phase et entre les phases. Elle s'effectue de la conception (ensemble de solutions) vers l'analyse (services d'information), et de l'analyse (services d'information) vers les besoins. La traçabilité s'applique aussi entre les éléments de la spécification des besoins et entre les éléments de l'analyse (service d'information), par exemple, entre les cas d'utilisation et les besoins exprimés sous forme textuelle. Les besoins doivent être identifiés conformément au § 7.3.2. Les résultats de la phase d'analyse spécifient, pour les différents cas d'utilisation, d'autres besoins en termes d'informations détaillées. La phase de conception doit pointer vers les divers diagrammes et textes figurant dans les résultats de la phase d'analyse. Le pointeur peut prendre la forme d'un renvoi vers les paragraphes correspondants.

En règle générale, la traçabilité de la phase de conception vers la phase d'expression des besoins métier se fait de façon indirecte, car les résultats de ces phases sont exprimés avec des niveaux de détail différents.

L'Annexe B fournit des lignes directrices pour la traçabilité entre la phase d'expression des besoins et la phase d'analyse.

Il est recommandé d'utiliser le mécanisme suivant pour procéder à la traçabilité avec les besoins, etc., spécifiés dans d'autres documents (qui ne suivent pas nécessairement le schéma d'identification préconisé):

plate-forme/organe "::" identifiant du document "::" id

où "id" correspond à l'une des valeurs suivantes:

1) identifiant du besoin;
2) identifiant du cas d'utilisation;
3) titre/texte du besoin;
4) titre du cas d'utilisation;
5) sous-paragraphe du document qui identifie, de façon unique, un besoin ou un cas d'utilisation.

Exemples:

3GPP::32.111-1::getAlarmList

UIT-T::M.3016::1.5.1.2

10 Structure de la documentation

Même si les phases sont au nombre de trois, leurs résultats peuvent être consignés dans un seul ou dans plusieurs documents de la documentation de l'interface. Il est néanmoins recommandé de consigner les besoins et l'analyse dans un même document et d'élaborer un document de conception pour chaque paradigme de protocole de gestion de réseau.

Annexe A

Expression des besoins

(Cette annexe fait partie intégrante de la présente Recommandation.)

La présente annexe décrit des lignes directrices pour la spécification des besoins. L'Appendice I illustre par un exemple l'utilisation de ce modèle.

Le modèle normal (ou complet) d'expression des besoins est décrit au § A.2. Un modèle simplifié figure au § A.3.

A.1 Conventions

A.1.1 Utilisation de la notation UML pour l'expression des besoins

Le Tableau A.1 donne la correspondance entre les concepts de gestion et la notation UML. La présente Recommandation spécifie les concepts et les notations de haut niveau à utiliser dans les différentes phases. Les stéréotypes sont utilisés pour étendre la notation UML. Les stéréotypes approuvés pour une utilisation dans l'environnement de gestion sont inclus dans la présente Recommandation (voir l'Annexe C).

Tableau A.1 – Concepts relatifs aux besoins

Concept relatif à la gestion	Notation UML	Remarque
utilisateur.	Acteur	Un utilisateur est modélisé comme un acteur.
rôle de gestion.	Acteur	Un acteur joue un rôle. Il est normalement conseillé de n'attribuer dans le modèle qu'un seul rôle par acteur.
fonction de gestion.	Cas d'utilisation	Une fonction de gestion est modélisée par un ou plusieurs cas d'utilisation.
ensemble de fonctions de gestion.	Cas d'utilisation	Un ensemble de fonctions de gestion est un cas d'utilisation composite dont chaque fonction de gestion est (éventuellement) modélisée par un cas d'utilisation distinct.
service de gestion.	Cas d'utilisation	Un service de gestion est modélisé par un cas d'utilisation de haut niveau.
scénario de gestion.	Diagramme de séquence	Il est préférable utiliser des diagrammes de séquence plutôt que des diagrammes de collaboration.
type de ressource de télécommunication.	Classe	Les diagrammes de classe décrivent les propriétés du type de ressource de télécommunication, avec un niveau de détail adapté à la phase de la méthodologie.
objectifs de gestion.	–	Aucune notation UML n'étant applicable, les objectifs de gestion sont consignés sous forme de descriptions textuelles.

A.1.2 Modèle de cas d'utilisation

Il convient de suivre les conventions et modèles suivants lors de l'élaboration des cas d'utilisation.

Tableau A.2 – Modèle de cas d'utilisation

Etape du cas d'utilisation	Evolution/Spécification	Utilisation de la relation <<utilise>>
Objectif(*)	Correspond à l'objectif/résultat final visé par le cas d'utilisation. L'objectif se présente sous la forme d'un énoncé concis qui décrit ce que le cas d'utilisation est censé réaliser dans un scénario réussi. On peut y associer un texte précisant la priorité du cas d'utilisation par rapport à d'autres cas d'utilisations ainsi que la qualité de fonctionnement attendue du cas d'utilisation, par exemple: • Temps réel. • Quasi-temps réel. • Non-temps réel.	
Acteurs et rôles(*)	Nom des acteurs/rôles intervenant dans le cas d'utilisation, y compris la caractéristique du rôle de chaque acteur.	
Ressources de télécommunication	Nom des ressources de télécommunication intervenant dans le cas d'utilisation.	
Hypothèses	Description de l'environnement dans lequel le cas d'utilisation s'applique. Les hypothèses et les préconditions s'excluent mutuellement. Les hypothèses concernent des propriétés statiques.	

Tableau A.2 – Modèle de cas d'utilisation

Etape du cas d'utilisation	Evolution/Spécification	Utilisation de la relation <<utilise>>
Préconditions	Liste de toutes les conditions que le système et l'environnement doivent remplir avant que le cas d'utilisation ne puisse être activé. Les préconditions et les hypothèses s'excluent mutuellement. Les préconditions sont liées à des propriétés dynamiques et peuvent donc aboutir à une exception, ce qui n'est jamais le cas avec les hypothèses.	
Commence lorsque	Nom de l'événement unique qui déclenche le scénario d'utilisation. Facultatif. N'est normalement pas utilisé pour spécifier des déclencheurs du type "lorsque le gestionnaire doit extraire des données".	
Etape 1(*) (M|O)	Un cas d'utilisation décrit la liste des étapes (manuelles ou automatiques) nécessaires pour réaliser son objectif. Les étapes peuvent faire appel à d'autres cas d'utilisation. Pour des besoins de traçabilité, les étapes sont numérotées. Pour chaque étape, on indique si elle est obligatoire (M pour *Mandatory*) ou optionnelle (O pour *Optional*). Les sous-étapes sont numérotées par rapport à l'étape de niveau supérieur, par exemple: Etape n Etape n.1 Etape n.2 où n.1 et n.2 sont des sous-étapes de l'étape n.	Référence à un cas d'utilisation utilisé.
Etape n (M|O)	Etapes ajoutées si nécessaire et selon un ordre logique.	
Se termine lorsque(*)	Evénement ou liste des événements indiquant que le cas d'utilisation est terminé. NOTE – Dans ce contexte, il convient de considérer le terme "événement" dans son sens le plus général, sans le limiter, par exemple, aux notifications échangées via une interface de gestion. A titre d'exemple, la fin d'un traitement peut être considérée comme un événement indiquant que le cas d'utilisation est terminé.	
Exceptions	Liste récapitulative de toutes les conditions d'exception et anomalies pouvant être détectées par le cas d'utilisation pendant son déroulement.	
Postconditions	Liste de toutes les conditions que le système et l'environnement doivent remplir une fois le cas d'utilisation terminé. L'énoncé des postconditions détermine si le cas d'utilisation doit être pleinement réussi, partiellement réussi, voire en échec pour considérer qu'il est terminé.	
Traçabilité(*)	Besoins ou cas d'utilisation exposés par le cas d'utilisation considéré.	
NOTE – Les champs marqués du signe "*" sont obligatoires pour toutes les spécifications de cas d'utilisation. Les autres champs ne sont obligatoires que lorsqu'ils sont pertinents pour le cas d'utilisation considéré.		

A.1.3 Catégories de besoins

Il est utile de classer les besoins en catégories. On considère que les catégories suivantes sont pertinentes pour la méthodologie MSIG:

– Conceptuel (CON) – Identifie un concept, un type de données, une relation, un format ou une structure.

– Fonctionnel (FON) – Identifie une capacité fonctionnelle, une situation dynamique, une séquence, des paramètres temporels ou une interaction.

– Non fonctionnel (NON) – Besoins non fonctionnels, notamment conditions anormales, conditions d'erreur et limites de performance.

– Administratif (ADM) – Besoins administratifs ou d'exploitation du système qui ne sont pas liés aux opérations normales des cas d'utilisation.

Les besoins doivent être rédigés en suivant le modèle ci-dessous:

REQ-Etiquette-Catégorie-Numéro {Catégorie, numéro} Détails {Source Citation}

où "Etiquette" est une abréviation correspondant à la Recommandation (ou à une partie de celle-ci). L'ensemble des étiquettes n'est ni limité ni normalisé.

Des lignes directrices pour la numérotation des besoins figurent à l'Appendice VI.

A.2 Modèle d'expression des besoins

1 Concepts et contexte

Définit les buts et objectifs principaux de cette spécification ainsi que les interfaces de gestion (et points de référence) applicables. Utilise les catégories définies dans [UIT-T M.3200] pour identifier le ou les services de gestion pris en charge par cette interface.

Ce paragraphe doit donner une description claire des avantages procurés aux utilisateurs, c'est-à-dire les raisons motivant l'exécution de ce service de gestion. Le contexte doit également être précisé si nécessaire, en veillant à séparer les parties explicatives des parties descriptives. Les informations contextuelles, le cas échéant, seront placées dans un appendice.

1.a SubClauseTitle

SubClauseTitle désigne le nom du sous-paragraphe.
"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque sous-paragraphe.
L'utilisation de sous-paragraphes est facultative.

2 Besoins de niveau métier

2.1 Besoins

2.1.a SubSetTitle
SubSetTitle désigne le nom d'un sous-ensemble des besoins de niveau métier.
"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque sous-ensemble.
L'utilisation des sous-ensembles est facultative, tous les besoins de niveau métier pouvant être exprimés au sous-paragraphe 2.1 (Besoins).
Enumérer les principaux besoins sous forme textuelle et identifier les différents cas d'utilisation en indiquant les acteurs/rôles et les ressources. Les cas d'utilisation de haut niveau (sous-paragraphe 2.4 ci-dessous) doivent faire ressortir les besoins de niveau métier. Ils se distinguent des besoins de niveau spécification par le fait qu'ils ne sont pas décomposés en niveaux inférieurs. Le paragraphe 2.4 contient

de nombreux exemples de cas d'utilisation de haut niveau. Les informations relatives aux différentes politiques (par exemple sécurité, persistance) se prêtent bien à une description à ce niveau. Pour des raisons de traçabilité, il est nécessaire de numéroter les besoins.

Les besoins doivent être spécifiés conformément au § A.1.3. Dans une spécification de besoins, il est proposé de consigner les besoins en suivant l'ordre décrit au § A.1.3 (pour l'ensemble de la spécification ou pour chaque sous-ensemble).

Si l'on décide d'utiliser les catégories de besoins – ce qui est facultatif –, on pourra n'en appliquer qu'un sous-ensemble.

A titre d'exemple, le besoin conceptuel n° 23 d'une Recommandation libellée "SM" serait spécifié comme suit:

Identifiant	Définition
REQ-SM-CON-23	Un Ordre de service consiste en un nom, une adresse, un numéro de téléphone, une description du service et un numéro de télécopie facultatif pour désigner les contacts {T1M1.5 Document 246 11/96}

On peut utiliser un ou plusieurs tableaux et des textes d'accompagnement entre les tableaux si nécessaire.

2.2 Rôles des acteurs

Ici figure une description textuelle de l'acteur (voir § 3).

2.3 Ressources de télécommunication

Ici figure une description textuelle des ressources de télécommunication requises (voir § 3) pour prendre en charge les cas d'utilisation.

2.4 Cas d'utilisation de haut niveau

On pourra insérer un diagramme de cas d'utilisation de haut niveau. Pour que les cas d'utilisation soient compris des spécialistes du domaine, ils doivent chacun être assortis d'une description textuelle. La description doit servir deux objectifs: d'une part, transcrire les connaissances des spécialistes du domaine, d'autre part, valider les modèles des phases d'analyse et de conception par rapport aux besoins. Un exemple de diagramme de cas d'utilisation de haut niveau est donné à l'Appendice I.

2.4.a UseCaseName

UseCaseName désigne le nom du cas d'utilisation.

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque nouvelle définition de cas d'utilisation.

Ce sous-paragraphe est répété pour chaque cas d'utilisation de haut niveau défini pour les besoins de niveau spécification de l'interface.

Les cas d'utilisation de haut niveau peuvent éventuellement identifier les différents ensembles de fonctions définis dans [UIT-T M.3400] ou les processus de gestion définis dans [UIT-T M.3050.x]. Ces cas d'utilisation peuvent être encore affinés comme décrit dans le sous-paragraphe "Besoins de niveau spécification" ci-après à l'aide de stéréotypes tels que "inclut" ("includes") et "étend" ("extends").

S'il y a lieu, on pourra utiliser des diagrammes de séquence. A noter toutefois qu'il n'est pas prévu d'utiliser ce type de diagramme pour les besoins de haut niveau. Ces diagrammes peuvent s'avérer plus utiles lorsque les cas d'utilisation décrits à ce niveau sont ensuite décomposés au niveau suivant de l'expression des besoins.

La traçabilité du niveau suivant par rapport à ce niveau peut être assurée par la façon dont chaque ensemble de fonctions est détaillé au moyen de nouveaux cas d'utilisation.

On pourra utiliser un ensemble de tableaux de cas d'utilisation (en suivant le modèle décrit au Tableau A.2) pour représenter les fonctionnalités pertinentes à un niveau d'abstraction adapté au problème analysé.

Le niveau de détail et la portée des cas d'utilisation sont subjectifs, car ils dépendent de la connaissance que l'équipe de rédaction des besoins a du domaine. Il y a toutes les chances pour que les niveaux de détail inférieurs soient une indication concernant l'analyse plutôt qu'une retranscription des besoins.

Il est autorisé d'analyser de plus en plus en détail chaque étape d'un cas d'utilisation de niveau d'abstraction supérieur en faisant référence, dans la cellule du tableau réservée à cet effet, au cas

d'utilisation détaillé. Il convient de souligner que cette façon de faire n'est pas une obligation et qu'elle dépend des besoins de l'auteur ou du groupe de rédaction.

Les questions suivantes sont destinées à faciliter la recherche des premiers cas d'utilisation:

– *Quelle est la finalité principale du système?*
– *Quels types de personnes/systèmes doivent interagir avec le système à l'étude?*
– *Comment ces personnes/systèmes peuvent-ils être regroupés ou synthétisés en rôles?*
– *Quelles sont les caractéristiques de démarrage, d'exploitation normale, de panne et de reprise après incident du système?*
– *Quels types de rapports ou de données provenant du système peuvent être nécessaires?*
– *Quelles activités spéciales sont requises (en fonction de l'heure ou de la charge du réseau par exemple)?*

Il est pertinent de documenter les cas d'utilisation de façon homogène. La structure suivante est proposée:

– *<tableau du cas d'utilisation> (voir le Tableau A.2)*
– *<diagramme(s) de séquence facultatif>*
– *<diagramme(s) d'états facultatif>*

3 Besoins de niveau spécification

3.1 Besoins

On l'a vu, les besoins de niveau métier sont détaillés au moyen des fonctions de gestion figurant la Recommandation dans [UIT-T M.3400]. Etant donné que cette Recommandation n'est pas suffisamment exhaustive pour couvrir tous les services de gestion pour tous les domaines gérés, il sera probablement nécessaire d'ajouter de nouvelles fonctions. Celles-ci devront être intégrées dans les besoins comme indiqué ci-dessous.

3.1.a SubSetTitle

SubSetTitle représente le nom d'un sous-ensemble de besoins de niveau spécification.

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque sous-ensemble.

L'utilisation des sous-ensembles est facultative, tous les besoins de niveau spécification pouvant être exprimés au sous-paragraphe 3.1 (Besoins).

Enumérer les principaux besoins détaillés et concrets sous forme textuelle et recenser les différents cas d'utilisation en indiquant les acteurs/rôles et les ressources. Par rapport aux cas d'utilisation de niveau métier, les cas d'utilisation figurant au sous-paragraphe 3.4 doivent décrire les besoins de niveau spécification en fournissant des informations de plus bas niveau, et être plus orientés vers l'implémentation. Pour des raisons de traçabilité, il est nécessaire de numéroter les besoins.

Les besoins doivent être spécifiés conformément au § A.1.3. Dans une spécification de besoins, il est proposé de consigner les besoins en suivant l'ordre décrit au § A.1.3 (pour l'ensemble de la spécification ou pour chaque sous-ensemble).

Si l'on décide d'utiliser les catégories de besoins – ce qui est facultatif –, on pourra n'en appliquer qu'un sous-ensemble.

A titre d'exemple, le besoin fonctionnel n° 33 d'une Recommandation libellée "OM" serait spécifié comme suit:

Identifiant	Définition
REQ-OM-FON-33	Une opération en attente peut être annulée par l'initiateur.

On peut utiliser un ou plusieurs tableaux avec des textes d'accompagnement entre les tableaux si nécessaire.

Les besoins de niveau spécification doivent suivre les conventions et les modèles définis au § A.1.

3.2 Rôles des acteurs

Ici figurent une liste de tous les acteurs ainsi qu'une description textuelle de tous les acteurs non définis plus haut dans les spécifications de niveau métier.

3.3 Ressources de télécommunication

Ici figurent une liste de toutes les ressources passives ainsi qu'une description textuelle de toutes les ressources non définies plus haut dans les besoins de niveau métier.

3.4 Cas d'utilisation

Les cas d'utilisation de haut niveau sont détaillés ici au moyen de plusieurs cas d'utilisation de niveau spécification, ces derniers faisant chacun l'objet d'une explication détaillée dans un sous-paragraphe, comme indiqué ci-dessous.

3.4.a UseCaseName

UseCaseName désigne le nom du cas d'utilisation.

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque nouvelle définition de cas d'utilisation.

On peut utiliser des diagrammes de séquence et des diagrammes d'états, s'il y a lieu.

NOTE – Les lignes directrices et les critères d'utilisation des diagrammes de séquence et des diagrammes d'états feront l'objet d'une étude ultérieure.

Les spécifications de cas d'utilisation doivent suivre les conventions et les modèles définis au § A.1.

A.3 Modèle simplifié de spécification des besoins

Le modèle simplifié de spécification des besoins est un modèle de substitution qui peut être utilisé lorsque seules les spécifications au format texte sont nécessaires. Un modèle distinct est défini ici pour éviter toute ambiguïté que causerait l'ajout d'options dans le modèle complet décrit au § A.2.

1 Concepts et contexte

Définit les buts et objectifs principaux de cette spécification ainsi que les interfaces de gestion (et points de référence) applicables. Utilise les catégories définies dans [UIT-T M.3200] pour identifier le ou les services de gestion pris en charge par cette interface.

Ce paragraphe doit donner une description claire des avantages procurés aux utilisateurs, c'est-à-dire les raisons motivant l'exécution de ce service de gestion. Le contexte doit également être précisé si nécessaire, en veillant à séparer les parties explicatives des parties descriptives. Les informations contextuelles, le cas échéant, seront placées dans un appendice.

1.a SubClauseTitle

SubClauseTitle désigne le nom du sous-paragraphe.

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque sous-paragraphe.

L'utilisation de sous-paragraphes est facultative.

2 Besoins

2.a SubSetTitle

SubSetTitle désigne le nom d'un sous-ensemble des besoins de niveau métier.

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque sous-ensemble.

L'utilisation des sous-ensembles est facultative, tous les besoins de niveau métier pouvant être exprimés au paragraphe 2 (Besoins).

Enumérer les principaux besoins sous forme textuelle et identifier les différents cas d'utilisation en indiquant les acteurs/rôles et les ressources. Les cas d'utilisation doivent faire ressortir les besoins de haut niveau. Ils se distinguent des besoins de niveau spécification par le fait qu'ils ne sont pas décomposés en niveaux inférieurs. Les informations relatives aux différentes politiques (par exemple sécurité, persistance) se prêtent bien à une description à ce niveau. Pour des raisons de traçabilité, il est nécessaire de numéroter les besoins.

Les besoins doivent être spécifiés conformément au § A.1.3. Dans une spécification de besoins, il est proposé de consigner les besoins en suivant l'ordre décrit au § A.1.3 (pour l'ensemble de la spécification ou pour chaque sous-ensemble).

Si l'on décide d'utiliser les catégories de besoins – ce qui est facultatif –, on pourra n'en appliquer qu'un sous-ensemble.

A titre d'exemple, le besoin conceptuel n^{o} 23 d'une Recommandation libellée "SM" serait spécifié comme suit:

Identifiant	Définition
REQ-SM-CON-23	Un Ordre de service consiste en un nom, une adresse, un numéro de téléphone, une description du service et un numéro de télécopie facultatif pour désigner les contacts {T1M1.5 Document 246 11/96}

On peut utiliser un ou plusieurs tableaux avec des textes d'accompagnement entre les tableaux si nécessaire.

Annexe B

Analyse

(Cette annexe fait partie intégrante de la présente Recommandation.)

Ci-après figurent des lignes directrices pour la spécification des résultats de la phase d'analyse.

Le modèle d'analyse a été élaboré à partir du service d'information du 3GPP [b-3GPP TS 32.151] et complété pour répondre aux exigences supplémentaires de la méthodologie (traçabilité par exemple).

Pour la spécification d'une interface de gestion, on utilisera les sous-paragraphes 2.2 et 2.3 du modèle d'analyse figurant au § B.2. Pour un modèle d'information (modèle de ressource de réseau par exemple), seul le sous-paragraphe 2.2 sera utilisé.

Le modèle d'analyse utilise le type d'information comme l'une des caractéristiques pour décrire les attributs des classes d'objets informationnels (IOC) ainsi que les paramètres d'opération/de notification. Le ou les types d'information valides pouvant être utilisés sont recensés à l'Annexe E avec leurs significations respectives.

L'Appendice II illustre par un exemple l'utilisation de ce modèle.

Les concepts "analyse|service d'information" et "conception|ensembles de solutions" désignent les spécifications équivalentes (mais nommées différemment) développées par l'UIT-T d'un côté et par le 3GPP de l'autre.

B.1 Conventions

B.1.1 Qualificatifs obligatoires, optionnels et conditionnels

Ce sous-paragraphe définit un certain nombre de termes utilisés pour qualifier la relation entre l'analyse (service d'information), la conception (ensembles de solutions) et leur incidence sur les implémentations des interfaces. Les qualificatifs définis dans ce sous-paragraphe servent uniquement à qualifier le comportement de l'agent. On considère que cela est suffisant pour la spécification des interfaces de gestion.

Les spécifications d'analyse (spécifications d'IS) définissent les attributs des classes IOC, les interfaces, les opérations, les notifications, les paramètres d'opération et les paramètres de notification. Elles peuvent avoir les qualificatifs "support/read/write" (pris en charge/lecture/écriture) suivants: M, O, CM, CO, C.

Définition du qualificatif M (obligatoire pour *Mandatory*):

- Utilisé pour les éléments qui doivent être pris en charge.

Définition du qualificatif O (optionnel pour *Optional*):

- Utilisé pour les éléments qui sont éventuellement mais pas nécessairement pris en charge.

Définition du qualificatif CM (conditionnel-obligatoire, pour *Conditional-Mandatory*):

- Utilisé pour les éléments qui sont obligatoires sous certaines conditions, plus précisément:
  - Pour tous les éléments possédant le qualificatif "support" CM, il faut définir une condition correspondante dans la Recommandation (spécification IS). Si la condition spécifiée est satisfaite, les éléments doivent être pris en charge.

Définition du qualificatif CO (conditionnel-optionnel pour *Conditional-Optional*):

- Utilisé pour les éléments qui sont optionnels sous certaines conditions, plus précisément:
  - Pour tous les éléments possédant le qualificatif "support" CO, il faut définir une condition correspondante dans la Recommandation (spécification IS). Si la condition spécifiée est satisfaite, les éléments peuvent éventuellement être pris en charge.

Définition du qualificatif C (ES-Conditionnel):

- Utilisé pour les éléments applicables à certaines conceptions (ensembles de solutions (ES)), mais pas à toutes.

Les spécifications de conception (ES) définissent les équivalents ES des attributs de classe IOC, des opérations, des notifications, des paramètres d'opération et des paramètres de notification. Ces équivalents ES peuvent être assortis des qualificatifs "support/read/write" (pris en charge/lecture/écriture) suivants: M, O, CM et CO.

La correspondance entre les qualificatifs des concepts d'analyse (concepts d'IS) et les qualificatifs des concepts ES est définie comme suit:

- Pour les qualificatifs M, O, CM et CO, chaque élément défini par un IS (opération et notification, paramètre d'opérations en entrée et en sortie, paramètre de notifications en

entrée, relation d'information et attribut d'information) sera mis en correspondance avec son ou ses équivalents dans tous les ES. Le ou les équivalents issus de la mise en correspondance auront le même qualificatif que celui défini par l'IS.

- Pour le qualificatif C, chaque élément défini par un IS sera mis en correspondance avec son ou ses équivalents dans un ES au moins. Le ou les équivalents issus de la mise en correspondance pourront avoir le qualificatif "support" M ou O.

Le Tableau B.1 définit la sémantique des qualificatifs des équivalents, en termes de prise en charge du point de vue de l'agent.

Tableau B.1 – Sémantique des qualificatifs utilisés dans la conception (ensembles de solutions)

Equivalent de l'ES issu de la mise en correspondance	Obligatoire	Optionnel	Conditionnel-Obligatoire (CM)	Conditionnel-Optionnel (CO)
Equivalent de la notification issu de la mise en correspondance	L'agent doit générer la notification.	L'agent peut éventuellement mais pas nécessairement générer la notification.	L'agent doit générer la notification si la condition associée à cet élément est satisfaite.	L'agent peut choisir ou non de générer la notification. S'il décide de la générer, la condition associée à cette notification doit être satisfaite.
Equivalent de l'opération issu de la mise en correspondance	L'agent doit prendre l'opération en charge.	L'agent peut éventuellement mais pas nécessairement prendre en charge l'opération. Si l'agent ne prend pas cette opération en charge, il doit rejeter l'appel de l'opération en indiquant pour motif "non pris en charge". Le rejet, accompagné d'un motif, doit être renvoyé au gestionnaire.	L'agent doit prendre en charge l'opération si la condition associée à cet élément est satisfaite.	L'agent peut éventuellement prendre en charge l'opération si la condition associée à cet élément est satisfaite.
Paramètre d'entrée de l'équivalent de l'opération issu de la mise en correspondance	L'agent doit accepter le paramètre et agir en fonction de sa valeur.	L'agent peut éventuellement mais pas nécessairement prendre en charge le paramètre d'entrée. S'il ne prend pas en charge ce paramètre d'entrée et que ce dernier achemine une information (autrement dit, il n'achemine pas une sémantique du type "pas d'information"), l'agent doit rejeter l'appel en invoquant un motif (motif = "paramètre non pris en charge"). Le rejet, accompagné du motif, doit être renvoyé au gestionnaire.	L'agent doit accepter le paramètre et agir en fonction de sa valeur si la condition liée à cet élément est satisfaite.	L'agent peut éventuellement accepter le paramètre et agir en fonction de sa valeur si la condition liée à cet élément est satisfaite.

Tableau B.1 – Sémantique des qualificatifs utilisés dans la conception (ensembles de solutions)

Equivalent de l'ES issu de la mise en correspondance	Obligatoire	Optionnel	Conditionnel-Obligatoire (CM)	Conditionnel-Optionnel (CO)
Paramètre d'entrée de l'équivalent de la notification issu de la mise en correspondance ET Paramètre de sortie de l'équivalent de l'opération issu de la mise en correspondance	L'agent doit fournir ce paramètre.	L'agent peut éventuellement fournir ce paramètre.	L'agent doit fournir ce paramètre si la condition associée à cet élément est satisfaite.	L'agent peut éventuellement fournir ce paramètre si la condition associée à cet élément est satisfaite.
Equivalent d'attribut de classe IOC issu de la mise en correspondance	L'agent doit prendre l'attribut en charge.	L'agent peut éventuellement prendre l'attribut en charge.	L'agent doit prendre en charge cet attribut si la condition associée à cet élément est satisfaite.	L'agent peut éventuellement prendre en charge cet attribut si la condition associée à cet élément est satisfaite.

B.2 Modèle d'analyse

1 Concepts et contexte

Ce paragraphe doit être une introduction à l'analyse des spécifications des interfaces de gestion.

1.a SubClauseTitle

SubClauseTitle correspond au nom du sous-paragraphe.

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque sous-paragraphe.

L'utilisation de sous-paragraphes est facultative.

2 Classes d'objets informationnels

Ce paragraphe doit être utilisé pour toutes les spécifications (spécifications des interfaces de gestion, mais aussi spécifications relatives à des modèles d'information seulement).

2.1 Entités d'information importées et étiquettes locales

Ce sous-paragraphe dresse la liste des entités d'information (par exemple, classe d'objets informationnels, interface, relation d'information, attribut d'information) qui ont été définies dans d'autres spécifications et qui sont importées dans le présent document. Toutes les entités importées doivent être traitées comme étant définies localement dans la présente spécification. On peut avoir besoin de réaliser une importation à des fins d'héritage. Chaque élément de cette liste est un couple (référence de l'étiquette, étiquette locale). La référence de l'étiquette contient le nom de la spécification à l'endroit où elle est définie, le type de l'entité d'information et son nom. L'étiquette locale des entités d'information importées peut ensuite être utilisée tout au long de la spécification en lieu et place de la référence de l'étiquette.

Ces informations sont consignées dans un tableau.

Référence de l'étiquette	Etiquette locale

Les éléments importés doivent être issus de définitions indépendantes du protocole reposant sur la présente méthodologie, mais ils peuvent aussi, si nécessaire, importer des éléments d'autres spécifications à des fins de migration, au fil du temps, de spécifications propres à certains protocoles.

On trouvera à l'Annexe F des lignes directrices sur l'importation des entités, sur les propriétés de classe IOC et sur l'héritage.

2.2 Diagramme de classes

2.2.1 Attributs et relations

Ce premier ensemble de diagrammes représente toutes les classes d'objets informationnels définies dans cet IS avec toutes leurs relations et tous leurs attributs, y compris les relations avec les IOC importées (le cas échéant). Ces diagrammes doivent comporter les cardinalités des classes d'objets informationnels (pour les associations et pour les relations de type conteneur) et peuvent aussi faire apparaître les noms des associations et des rôles. Ils doivent être des diagrammes de classes conformes au langage UML (voir également l'Annexe C).

Il n'est pas nécessaire de répéter, dans le diagramme, les caractéristiques (relations) des classes d'objets informationnels importées. Les classes d'objets informationnels doivent être définies à l'aide du stéréotype <<InformationObjectClass>>.

2.2.2 Héritage

Ce deuxième ensemble de diagrammes représente la hiérarchie d'héritage de toutes les classes d'objets informationnels définies dans cet IS. Il n'est pas nécessaire d'intégrer dans ces diagrammes la hiérarchie d'héritage complète, mais au minimum les classes d'objets informationnels parentes de toutes les classes d'objets informationnels définies dans le présent document. Par défaut, une classe d'objets informationnels hérite de la classe d'objets informationnels "top". Ces diagrammes doivent être des diagrammes de classes conformes au langage UML.

Il n'est pas nécessaire de répéter, dans le diagramme, les caractéristiques (attributs, relations) des classes d'objets informationnels importées. Les classes d'objets informationnels doivent être définies à l'aide du stéréotype <<InformationObjectClass>>.

NOTE 1 – Pour une meilleure lisibilité, certaines relations d'héritage présentées au sous-paragraphe 2.2.2 peuvent être répétées au sous-paragraphe 2.2.1.

NOTE 2 – L'héritage des interfaces n'est pas présenté dans ce sous-paragraphe (2.2.2), mais au sous-paragraphe 3.1.

2.3 Définitions des classes d'objets informationnels

Chaque classe d'objets informationnels est définie au moyen de la structure suivante.

Les éléments hérités (attributs, etc.) ne doivent pas être indiqués, étant donné qu'ils sont définis dans le ou les IOC parentes et qu'ils sont donc valides pour toutes les sous-classes.

2.3.a InformationObjectClassName

InformationObjectClassName correspond au nom de la classe d'objets informationnels.

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque nouvelle définition d'IOC.

2.3.a.1 Définition

Le sous-paragraphe <Définition> est rédigé en langage naturel et fait référence à la classe d'objets informationnels elle-même. Les caractéristiques relatives aux relations que la classe d'objets peut avoir avec d'autres classes d'objets ne peuvent pas se trouver dans la définition. Pour trouver ce type d'information, le lecteur doit se reporter à la définition des relations. C'est ici en revanche que les informations d'héritage doivent être spécifiées.

Les informations de traçabilité permettant de remonter à un ou plusieurs besoins pris en charge par cette IOC doivent aussi être indiquées dans ce sous-paragraphe, sous la forme suivante:

Référence	Etiquette du besoin	Remarque

2.3.a.2 Attributs

Le sous-paragraphe <Attributs> dresse la liste des attributs, qui sont les propriétés gérables de la classe d'objets. Chaque élément est représenté par un n-uplet (attributeName, supportQualifier, readQualifier, writeQualifier):

- *supportQualifier indique si l'attribut est obligatoire (M,* Mandatory*), optionnel (O,* Optional*), conditionnel-obligatoire (CM,* Conditional-Mandatory*), conditionnel-optionnel (CO,* Conditional-Optional*), ES-Conditionnel (C) ou non pris en charge (–). Les valeurs autorisées sont: "obligatoire", "optionnel", "conditionnel" ou "non pris en charge" (respectivement "M", "O", "C" et "–").*
- *readQualifier indique si l'attribut peut être lu par le gestionnaire. Il peut prendre les valeurs suivantes: obligatoire (M,* Mandatory*), optionnel (O,* Optional*), conditionnel-obligatoire (CM,* Conditional-Mandatory*), conditionnel-optionnel (CO,* Conditional-Optional*), ES-Conditionnel (C) ou Non pris en charge (–). Les valeurs autorisées sont: "obligatoire (M)", "conditionnel (C)" et "non pris en charge (–)".*
- *writeQualifier indique si l'attribut est accessible en écriture par le gestionnaire. La sémantique de writeQualifier est identique à celle de supportQualifier pour "M", "O" et "– ". Les valeurs autorisées sont: "obligatoire (M)", "conditionnel (C)" et "non pris en charge (–)".*

Il existe une relation de dépendance entre les qualificatifs supportQualifier, readQualifier et writeQualifier. Le qualificatif supportQualifier indique s'il est nécessaire de prendre en charge l'attribut. Quel que soit l'attribut, indépendamment de la valeur de supportQualifier, l'un au moins des qualificatifs readQualifier et writeQualifier doit avoir la valeur "M". La valeur "O" du qualificatif supportQualifier a pour effet de rendre l'attribut optionnel; si celui-ci est effectivement pris en charge, les qualificatifs read et write indiquent comment il doit l'être.

Par définition, les attributs privés ou internes à l'agent ne sont pas accessibles en écriture par le gestionnaire du point de référence d'intégration (IRPManager). Leur qualificatif writeQualifier est donc toujours égal à "–".

Les qualificatifs readQualifier et writeQualifier d'un attribut public pris en charge ne peuvent pas être tous les deux égaux à "–".

L'utilisation de "–" dans le qualificatif supportQualifier est réservée à la documentation de la prise en charge des attributs définis par une IOC "Archétype" (voir le sous-paragraphe C.3.5). Les attributs dont le qualificatif supportQualifier est égal à "–" ne sont pas implémentés par les IOC qui ne réalisent qu'un sous-ensemble des attributs définis par l'"Archétype". Dans ce cas, les qualificatifs readQualifier et writeQualifier ne sont pas pertinents. Cela étant, un attribut non pris en charge ne peut être ni lu ni accessible en écriture. Les qualificatifs readQualifier et writeQualifier doivent donc être égaux à "–" lorsque l'attribut n'est pas pris en charge.

Pour toute IOC utilisant un ou plusieurs attributs d'un "Archétype", un tableau distinct doit indiquer les attributs pris en charge. Si aucun attribut d'"Archétype" n'est pris en charge, ce tableau n'existe pas. Par exemple, si une IOC donnée possède des attributs propres (c'est-à-dire des attributs non définis par un "Archétype") et qu'elle encapsule les attributs de deux "Archétypes", alors la totalité des attributs de l'IOC en question figurera dans trois tableaux distincts.

Ces informations sont consignées dans un tableau.

Nom de l'attribut	Qualificatif de prise en charge (*Support qualifier*)	Qualificatif de lecture (*Read qualifier*)	Qualificatif d'écriture (*Write qualifier*)	Identifiants des besoins

2.3.a.3 Contraintes d'attributs

Le sous-paragraphe <Contraintes d'attributs> présente les contraintes entre attributs qui sont toujours considérées comme vraies. Ces propriétés sont toujours considérées comme vraies pendant la durée de vie des attributs. En particulier, il n'est pas nécessaire de les répéter dans les préconditions ou les postconditions des opérations ou des notifications.

NOTE – Il n'est pas nécessaire d'inclure ce sous-paragraphe lorsqu'il n'y a pas de contraintes d'attributs à définir.

2.3.a.4 Relations

Le sous-paragraphe <Relations> dresse la liste des relations auxquelles cette classe participe. Chaque élément est un relationshipName (nom de relation).

Les relations sont énumérées dans un tableau, comme indiqué ci-dessous:

Relation	Identifiants des besoins

Chaque nom de relation doit être un renvoi (de préférence également un hyperlien) vers le sous-paragraphe approprié du § 2 (Classes d'objets informationnels).

NOTE – Ce sous-paragraphe est facultatif, car toutes les relations sont représentées dans le diagramme de classes figurant au sous-paragraphe 2.2.1.

2.3.a.5 Diagramme d'états

Le sous-paragraphe <Diagramme d'états> contient des diagrammes d'états. Le diagramme d'états d'une classe d'objets informationnels définit les états autorisés de cette classe ainsi que les transitions entre ces états. Un état peut être exprimé sous plusieurs formes: valeurs d'attributs, combinaison de valeurs d'attributs ou participation aux relations de la classe d'objets informationnels en cours de définition. Ce diagramme doit être un diagramme d'états conforme au langage UML.

NOTE – Il n'est pas nécessaire d'inclure ce sous-paragraphe lorsqu'il n'y a pas de diagramme d'états à définir.

2.3.a.6 Notifications

Pour cette IOC, le sous-paragraphe <Notifications>, présente:

a) à titre facultatif, une référence aux notifications communes définies au sous-paragraphe 2.6 qui sont valides pour cette IOC; et

b) à titre facultatif, une liste de notifications à exclure de la liste des notifications communes (figurant au sous-paragraphe 2.6) pour cette IOC (à noter: les notifications héritées du ou des IOC parentes ne peuvent pas être exclues); et

c) à titre facultatif, la liste des notifications applicables à cette IOC et qui sont éventuellement mais pas nécessairement définies dans les notifications communes figurant au sous-paragraphe 2.6.

Les notifications identifiées dans ce sous-paragraphe sont celles qui peuvent être émises via l'interface de gestion, les paramètres "classe de l'objet" et "instance de l'objet" de leur en-tête (voir la Note 2) déterminant une instance de l'IOC définie par le sous-paragraphe de niveau supérieur (à savoir, le sous-paragraphe 2.3.a).

Les notifications identifiées dans ce sous-paragraphe peuvent provenir d'un ou plusieurs objets d'implémentation dont l'identifiant est mappé, dans l'implémentation, sur l'identifiant de l'instance de l'objet utilisé dans l'interface de gestion. Par conséquent, la présence de notifications dans ce sous-paragraphe (à savoir 2.3.a.6) ne signifie pas nécessairement qu'elles proviennent d'une instance de l'IOC définie par le sous-paragraphe de niveau supérieur (2.3.a) et ne les identifie pas comme telles.

Les informations relatives à l'option c) précitée sont reportées dans un tableau, qui se présentera par exemple comme suit:

Nom	Qualificatif	Identifiants des besoins	Remarques

NOTE 1 – Ce sous-paragraphe et ce tableau peuvent être absents.

NOTE 2 – L'en-tête de notification est défini dans le service d'information de point IRP de notification [b-3GPP TS 32.302].

NOTE 3 – Le qualificatif d'une notification figurant dans le tableau des notifications indique si une notification peut acheminer le DN (nom distinctif) de l'instance à l'intérieur de la notification. Le qualificatif d'une notification précisé dans une spécification de gestion indique le niveau de prise en charge relatif à l'émission de la notification du sujet (subject).

Un gestionnaire peut recevoir une notification XYZ contenant le DN de l'instance d'une classe ABC si et seulement si:

1) *La notification XYZ figure dans le tableau des notifications de la classe ABC; et*
2) *L'implémentation de l'instance de la classe ABC prend en charge la notification XYZ; et*
3) *Une interface de gestion définit la notification XYZ; et*
4) *L'implémentation de l'interface de gestion prend en charge la notification XYZ.*

2.4 Définitions des relations d'information

Dans un premier temps, ce sous-paragraphe énumère, dans le tableau suivant, toutes les relations prises en charge par cette Recommandation (Spécification). Le qualificatif de prise en charge est défini de la même manière que les attributs énumérés au § B.1.

Relation	Qualificatif de prise en charge (Support qualifier)	Identifiants des besoins

Chaque relation d'information est définie au moyen de la structure suivante.

Les relations héritées ne doivent pas être indiquées, étant donné qu'elles sont définies par le ou les IOC parentes et qu'elles sont donc valides pour toutes les sous-classes.

2.4.a InformationRelationshipName (supportQualifier)

InformationRelationshipName désigne le nom de la relation d'information suivi d'un qualificatif (voir le § B.1).

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque nouvelle définition de relation d'information.

2.4.a.1 Définition

Le sous-paragraphe <Définition> est rédigé en langage naturel.

2.4.a.2 Rôles

Le sous-paragraphe <Rôles> fournit les rôles joués par les classes d'objets dans la relation. Chaque élément est un couple (roleName, roleDefinition).

Ces informations sont consignées dans un tableau.

Nom	Définition

2.4.a.3 Contraintes

Le sous-paragraphe <Contraintes> dresse la liste des propriétés qui précisent les invariants sémantiques à conserver dans la relation. Chaque élément est un couple (propertyName, propertyDefinition). Ces propriétés étant toujours considérées comme vraies pendant la durée de vie de la relation, il n'est pas nécessaire de les répéter dans les préconditions ou les postconditions des opérations ou des notifications.

Ces informations sont consignées dans un tableau.

Relation	Qualificatif de prise en charge (*Support qualifier*)	Identifiants des besoins

2.5 Définitions des attributs d'information

Chaque attribut d'information est défini au moyen de la structure suivante.

Les attributs hérités ne doivent pas être indiqués, étant donné qu'ils sont définis dans le ou les IOC parentes et qu'ils sont donc valides pour toutes les sous-classes.

2.5.1 Définition et valeurs autorisées

Ce sous-paragraphe contient, pour chaque attribut à définir, son nom, sa définition en langage naturel, son type d'information (voir l'Annexe E) et une liste facultative de ses valeurs autorisées.

Si les valeurs autorisées peuvent être énumérées, chaque élément est un couple (Legal Value Name, Legal Value Semantics) [nom de la valeur autorisée, sémantique de la valeur autorisée], sauf si une sémantique de valeur autorisée s'applique à plusieurs valeurs, auquel cas elle n'est indiquée qu'une seule fois. Si les valeurs autorisées ne peuvent pas être énumérées, la liste des valeurs autorisées fait l'objet d'une seule et même définition.

Ces informations sont consignées dans un tableau.

Nom de l'attribut	Définition	Type d'information/ Valeurs autorisées

2.5.2 Contraintes

Le sous-paragraphe <Contraintes> indique s'il existe des contraintes sur les attributs. Chaque contrainte est définie par un n-uplet (propertyName, affected attributes, propertyDefinition) [nom de la propriété, attributs concernés, définition de la propriété]. Les éléments PropertyDefinition sont exprimés en langage naturel.

Ces informations sont consignées dans un tableau.

Nom	Attribut(s) concerné(s)	Définition

2.6 Notifications communes

Le sous-paragraphe <Notifications communes> dresse la liste des notifications qui peuvent être référencées par toute classe IOC définie par cette spécification d'interface de gestion. Ces notifications ne s'appliquent qu'aux IOC faisant référence à ce sous-paragraphe dans le sous-paragraphe 2.3.a.6.

Ces informations sont consignées dans un tableau.

Nom	Qualificatif	Remarques

NOTE – Il n'est pas nécessaire de faire figurer ce sous-paragraphe lorsqu'il n'y a pas de notifications communes.

2.7 Modèle d'états du système

Certaines configurations d'information sont suffisamment spéciales ou complexes pour qu'il soit justifié de recourir à un diagramme d'états pour les clarifier. Dans ce sous-paragraphe, un diagramme d'états définit les états autorisés du système ainsi que les transitions entre ces états. Un état est exprimé sous la forme d'une combinaison de valeurs d'attribut ou de participation à des relations mettant en jeu une ou plusieurs classes d'objets informationnels.

3 Définition des interfaces

Ce paragraphe doit être utilisé pour toutes les spécifications des interfaces de gestion. Il est facultatif pour les spécifications relatives à des modèles d'information seulement.

3.1 Diagramme de classes représentant les interfaces

Chaque interface est définie dans le diagramme, qui doit être un diagramme de classes conforme au langage UML (voir aussi l'Annexe C).

Les interfaces sont définies à l'aide d'un stéréotype <<Interface>>. Chaque interface contient un ensemble composé: soit d'opérations ou de notifications obligatoires, soit d'une seule opération ou d'une seule notification obligatoire. Les stéréotypes (voir l'Annexe C) servent à spécifier les interfaces facultatives ou obligatoires. Dans le diagramme de classes, chaque opération et notification d'une interface doit être qualifiée de "publique" par l'ajout d'un symbole "+" avant chaque opération et notification.

NOTE – Il est possible de faire apparaître l'héritage d'interfaces dans ce sous-paragraphe.

3.2 Règles génériques

Les règles suivantes valent pour toutes les spécifications. Il suffira de les recopier pour les intégrer à la spécification considérée.

Règle 1: Chaque opération possédant au moins un paramètre d'entrée est assortie d'une précondition valid_input_parameter indiquant que tous les paramètres d'entrée doivent être valides au regard de leur type d'information. De plus, toutes les opérations de ce type sont assorties d'une exception operation_failed_invalid_input_parameter qui est levée lorsque la précondition valid_input_parameter est fausse. Les états d'entrée et de sortie de l'exception sont identiques.

Règle 2: Chaque opération possédant au moins un paramètre d'entrée optionnel est assortie d'un ensemble de préconditions supported_optional_input_parameter_xxx où "xxx" correspond au nom du paramètre d'entrée optionnel, la précondition indiquant que l'opération prend en charge le paramètre d'entrée optionnel désigné. De plus, toutes les opérations de ce type sont assorties d'une exception operation_failed_unsupported_optional_input_parameter_xxx qui est levée lorsque a) la précondition supported_optional_input_parameter_xxx est fausse et b) le paramètre d'entrée optionnel désigné achemine de l'information. Les états d'entrée et de sortie de l'exception sont identiques.

Règle 3: Chaque opération doit prendre en charge une exception générique operation_failed_internal_problem, qui est levée lorsqu'un problème interne survient et que l'opération ne peut pas être menée à terme. Les états d'entrée et de sortie de l'exception sont identiques.

NOTE – Les considérations touchant à la sécurité et les règles génériques qui en résultent feront l'objet d'une étude ultérieure.

3.b Interface InterfaceName (supportQualifier)

InterfaceName est le nom de l'interface suivi d'un qualificatif (voir le §B.1).

"b" est un numéro, qui commence à 3 et est incrémenté de 1 à chaque nouvelle définition d'interface.

Chaque interface est définie par son nom et par une séquence d'opérations ou de notifications comme indiqué ci-dessous.

Chaque opération est définie au moyen de la structure suivante.

NOTE – Le regroupement d'opérations/partitionnement des contenus d'interface et la dénomination des interfaces feront l'objet d'une étude ultérieure.

3.b.a Opération OperationName (supportQualifier)

OperationName est le nom de l'opération suivi d'un qualificatif (voir le § B.1).

"a" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque nouvelle définition d'opération.

3.b.a.1 Définition

Le sous-paragraphe <Définition> est rédigé en langage naturel.

Les informations de traçabilité permettant de remonter à un ou plusieurs besoins pris en charge par cette opération doivent aussi être indiquées dans ce sous-paragraphe, sous la forme suivante:

Référence	Etiquette du besoin	Remarque

3.b.a.2 Paramètres d'entrée

Liste des paramètres d'entrée de l'opération. Chaque élément est un n-uplet (Parameter Name [nom du paramètre], Support Qualifier [qualificatif de prise en charge], Information Type [type d'information] (voir l'Annexe E et la Note du § E.2), liste facultative de valeurs autorisées prises en charge par le paramètre, Comment [remarques]). Les valeurs autorisées du qualificatif de prise en charge sont spécifiées au § B.1.

Ces informations sont consignées dans un tableau.

Nom du paramètre	Qualificatif de prise en charge (*Support qualifier*)	Type d'information de correspondance/ Valeurs autorisées	Remarque

NOTE – Le Type d'information qualifie le paramètre de nom "Nom du paramètre". Si les valeurs autorisées peuvent être énumérées, chaque élément est un couple (Legal Value Name, Legal Value Semantics) [nom de la valeur autorisée, sémantique de la valeur autorisée], sauf si une sémantique de valeur autorisée s'applique à plusieurs valeurs, auquel cas la définition n'est indiquée qu'une seule fois. Si les valeurs autorisées ne peuvent pas être énumérées, la liste des valeurs autorisées fait l'objet d'une seule et même définition.

3.b.a.3 Paramètres de sortie

Liste des paramètres de sortie de l'opération. Chaque élément est un n-uplet (Parameter Name [nom du paramètre], Support Qualifier [qualificatif de prise en charge], Matching Information/Information Type [information de correspondance/type d'information] (voir l'Annexe E et la Note du § E.2), liste facultative de valeurs autorisées prises en charge par le paramètre, Comment [remarque]). Les valeurs autorisées du qualificatif de prise en charge sont spécifiées au § B.1.

Ces informations sont consignées dans un tableau.

Nom du paramètre	Qualificatif de prise en charge (*Support qualifier*)	Information de correspondance/ Type d'information/ Valeurs autorisées	Remarque

NOTE – Le Type d'information qualifie le paramètre de nom "Nom du paramètre". Si les valeurs autorisées peuvent être énumérées, chaque élément est un couple (Legal Value Name, Legal Value Semantics) [nom de la valeur autorisée, sémantique de la valeur autorisée], sauf si une sémantique de valeur autorisée s'applique à plusieurs valeurs, auquel cas la définition n'est indiquée qu'une seule fois. Si les valeurs autorisées ne peuvent pas être énumérées, la liste des valeurs autorisées fait l'objet d'une seule et même définition.

Ce tableau doit aussi contenir un paramètre spécial appelé "statut", qui correspond au statut d'exécution de l'opération (réussie, partiellement réussie, dysfonctionnement, etc.).

3.b.a.4 Précondition

Une précondition est un ensemble d'affirmations reliées par les opérateurs logiques ET, OU et NON. La précondition doit être vraie avant que l'opération ne soit invoquée.

Chaque affirmation est un couple (propertyName [nom de la propriété], propertyDefinition [définition de la propriété]). Toutes les affirmations constituant la précondition sont consignées dans un tableau.

Nom de l'affirmation	Définition

3.b.a.5 Postconditions

Une postcondition est un ensemble d'affirmations reliées par les opérateurs logiques ET, OU et NON. La postcondition doit être vraie après l'exécution de l'opération. Lorsqu'une postcondition ne précise rien en ce qui concerne une entité d'information, on considère que cette entité n'a pas été modifiée par rapport à ce qui est énoncé dans la précondition.

Chaque affirmation est un couple (propertyName [nom de la propriété], propertyDefinition [définition de la propriété]). Toutes les affirmations constituant la postcondition sont consignées dans un tableau.

Nom de l'affirmation	Définition

3.b.a.6 Exceptions

Liste des exceptions qui peuvent être levées par l'opération. Chaque élément est un n-uplet (exceptionName [nom de l'exception], condition [condition], ReturnedInformation [information renvoyée], exitState [état de sortie]).

3.b.a.6.c exceptionName

ExceptionName désigne le nom de l'exception.

"c" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque nouvelle définition d'exception.

Ces informations sont consignées dans un tableau.

Nom de l'exception	Définition	
	Condition	
	Information renvoyée	
	Etat de sortie	
	Condition	
	Information renvoyée	
	Etat de sortie	

3.b.a.7 Contraintes

Le sous-paragraphe <Contraintes> présente les contraintes applicables à l'opération ou à ses paramètres.

NOTE – Il n'est pas nécessaire d'inclure ce sous-paragraphe lorsqu'il n'y a pas de contraintes à définir.

3.b.b Notification NotificationName (supportQualifier)

NotificationName désigne le nom de la notification suivi d'un qualificatif (voir le § B.1).

"b" est un numéro, qui commence à 1 et est incrémenté de 1 à chaque nouvelle définition de notification.

3.b.b.1 Définition

Le sous-paragraphe <Définition> est rédigé en langage naturel.

Les informations de traçabilité permettant de remonter à un ou plusieurs besoins pris en charge par cette notification doivent aussi être indiquées dans ce sous-paragraphe, sous la forme suivante:

Référence	Etiquette du besoin	Remarque

3.b.b.2 Paramètres d'entrée

Liste des paramètres d'entrée de la notification. Chaque élément est un n-uplet (Parameter Name [nom du paramètre], Qualifiers [qualificatifs], Matching Information/Information Type [information de correspondance/type d'information] (voir l'Annexe E et la Note du § E.2), liste facultative de valeurs autorisées prises en charge par le paramètre, Comment [remarque]).

La colonne "qualificatifs" contient les deux qualificatifs Support Qualifier [qualificatif de prise en charge] (voir le § B.1) et Filtering Qualifier [qualificatif de filtrage], séparés par une virgule. Le qualificatif de filtrage indique si le paramètre de la notification peut être filtré ou non. Il peut prendre deux valeurs: Oui (O) ou Non (N).

Ces informations sont consignées dans un tableau.

Nom du paramètre	Qualificatifs	Information de correspondance/ Type d'information/ Valeurs autorisées	Remarque

NOTE – Le Type d'information qualifie le paramètre de nom "Nom du paramètre". Si les valeurs autorisées peuvent être énumérées, chaque élément est un couple (Legal Value Name, Legal Value Semantics) [nom de la valeur autorisée, sémantique de la valeur autorisée], sauf si une sémantique de valeur autorisée s'applique à plusieurs valeurs, auquel cas la définition n'est indiquée qu'une seule fois. Si les valeurs autorisées ne peuvent pas être énumérées, la liste des valeurs autorisées fait l'objet d'une seule et même définition.

3.b.b.3 Evénement de déclenchement

L'événement de déclenchement de la notification à envoyer correspond à la transition de l'état d'information défini par le sous-paragraphe "état d'origine" vers l'état d'information défini par le sous-paragraphe "état de destination".

3.b.b.3.1 Etat d'origine

Ce sous-paragraphe est un ensemble d'affirmations reliées par les opérateurs logiques ET, OU et NON.

Chaque affirmation est un couple (propertyName [nom de la propriété], propertyDefinition [définition de la propriété]). Toutes les affirmations constituant l'état d'origine sont consignées dans un tableau.

Nom de l'affirmation	Définition

3.b.b.3.2 Etat de destination

Ce sous-paragraphe est un ensemble d'affirmations reliées par les opérateurs logiques ET, OU et NON. Lorsqu'un état de destination ne précise rien en ce qui concerne une entité d'information, on considère que cette entité n'a pas été modifiée par rapport à ce qui est énoncé dans l'état d'origine.

Chaque affirmation est un couple (propertyName [nom de la propriété], propertyDefinition [définition de la propriété]). Toutes les affirmations constituant l'état de destination sont consignées dans un tableau.

Nom de l'affirmation	Définition

3.b.b.4 Contraintes

Le sous-paragraphe <Contraintes> présente les contraintes applicables à la notification ou à ses paramètres.

NOTE – Il n'est pas nécessaire d'inclure ce sous-paragraphe lorsqu'il n'y a pas de contraintes à définir.

3.c Scénario

Ce sous-paragraphe contient un ou plusieurs diagrammes de séquence, chacun décrivant un scénario possible. Ces diagrammes doivent être des diagrammes de séquence conformes au langage UML. Ce sous-paragraphe est facultatif.

B.3 Propriétés des classes IOC, héritage et importation

B.3.1 Propriétés

Les propriétés des classes IOC (à l'exception des classes IOC de prise en charge (*Support IOC*)) sont spécifiées sur la base des informations suivantes:

a) Un ou plusieurs attributs de classe IOC, y compris sa sémantique et sa syntaxe, ses intervalles de valeurs autorisées et ses qualificatifs de prise en charge. Les attributs de classe IOC ne se limitent pas à la gestion de la configuration; ils incluent aussi ceux concernant, par exemple, 1) la gestion de la qualité de fonctionnement (à savoir, les types de mesure), 2) la gestion des traces et 3) la gestion de la comptabilisation.

b) Le comportement, non spécifique à un attribut, associé à une classe IOC (voir la Note 1).

NOTE 1 – Par exemple, le lien entre A et B est optionnel. Il est obligatoire si l'instance A appartient à une instance de ManagedElement et que l'instance B appartient à une autre instance de ManagedElement. Ce comportement "lien" est un comportement non spécifique à un attribut. En principe, ce comportement, comme d'autres, sera hérité.

c) Une ou plusieurs relations de la classe IOC avec une ou plusieurs autres IOC.

d) Un ou plusieurs types de notification de la classe IOC et leurs qualificatifs.

e) La relation d'une classe IOC avec ses parents (voir la Note 2). On dénombre trois cas qui s'excluent mutuellement:

1) La classe IOC est abstraite et aucun parent n'a encore été désigné.

2) La classe IOC est abstraite, tous les parents possibles (un seul éventuellement) ont été désignés et les sous-classes IOC peuvent être désignées comme une classe IOC racine.

3) La classe IOC n'est pas abstraite, tous les parents possibles (un seul éventuellement) ont été désignés et la classe IOC peut être désignée comme une classe IOC racine.

   De deux choses l'une: soit l'instance d'une classe IOC est une classe IOC racine, soit elle possède un parent et un seul.

NOTE 2 – La relation parent-enfant dans ce sous-paragraphe correspond à la relation "nom du parent-contient l'enfant".

f) La relation d'une classe IOC avec ses enfants. On dénombre trois cas qui s'excluent mutuellement:

1) Une classe IOC ne peut pas avoir de classes IOC enfants (relation nom-conteneur).

2) Une classe IOC peut avoir une ou plusieurs classes IOC enfants. Le nombre maximal d'instances par classe IOC enfant peut être spécifié. Une classe IOC peut indiquer que les objets propres à un fabricant ne peuvent pas être des classes IOC enfants.

3) Une classe IOC ne peut avoir comme enfants que certaines classes IOC (ou leurs sous-classes). Le nombre maximal d'instances par classe IOC enfant peut être spécifié. Une classe IOC peut indiquer que les objets propres à un fabricant ne peuvent pas être des classes IOC enfants.

g) Information indiquant si une classe IOC peut être instanciée ou non (autrement dit, si une classe IOC est une classe abstraite).

h) Un attribut à des fins de dénomination.

B.3.2 Héritage

Une classe IOC (la sous-classe) qui hérite d'une autre classe IOC (la superclasse) possède toutes les propriétés de la superclasse.

La sous-classe peut modifier la ou les qualificatifs de prise en charge (*support-qualifications*) hérités qui sont optionnels de façon à les rendre obligatoires, mais pas l'inverse. La sous-classe peut modifier un qualificatif de prise en charge hérité qui est conditionnel-optionnel de façon à le rendre conditionnel-obligatoire, mais pas l'inverse.

Une classe IOC peut être la superclasse de nombreuses classes IOC. Une sous-classe ne peut pas avoir plus d'une superclasse.

La sous-classe peut:

a) Ajouter des attributs qui lui sont propres (par rapport à ceux de sa superclasse), y compris leur comportement, leurs intervalles de valeurs autorisées et leurs qualificatifs de prise en charge. Chaque attribut supplémentaire doit posséder un nom unique (parmi tous les attributs ajoutés et hérités).

b) Ajouter un comportement non spécifique à un attribut sur la base d'une classe IOC. Ce comportement ne doit pas entrer en contradiction avec le comportement de la superclasse héritée.

c) Ajouter une ou plusieurs relations avec une ou plusieurs classes IOC. Chaque relation supplémentaire doit posséder un nom unique (parmi toutes les relations ajoutées et héritées).

d) Ajouter des types de notification supplémentaires et leurs qualificatifs.

e) Désigner tous les parents possibles (éventuellement un seul) (ainsi que leurs sous-classes) si la superclasse possède la propriété-e-1, de sorte qu'une classe IOC possédera la propriété-e-2 ou la propriété-e-3. Restreindre le(s) parent(s) possible(s) (et leurs sous-classes) et/ou supprimer la possibilité pour la sous-classe d'être une classe IOC racine si la superclasse possède la proriété-e-2 ou la propriété-e-3.

f) Ajouter une ou plusieurs classes IOC enfants si la superclasse possède la propriété-f-2 de sorte qu'une classe IOC possédera la propriété-f-3. Restreindre les classes IOC enfants autorisées (ou leurs sous-classes) si la superclasse possède la propriété-f-3.

g) Préciser si une classe IOC peut être instanciée ou non (autrement dit, si la classe IOC est une classe abstraite).

h) Limiter l'intervalle des valeurs autorisées d'un attribut d'une superclasse possédant un tel intervalle.

B.3.3 Importation

Pour faciliter la réutilisation de définitions de classe IOC au sein de spécifications de point IRP, un mécanisme d'importation est utilisé: une spécification de point IRP (appelé le "point IRP sujet" (*subject IRP*)) peut avoir recours à ce mécanisme pour réutiliser la définition de classes IOC figurant dans une autre spécification de point IRP. Lorsque la spécification du point IRP sujet importe une classe IOC, elle ne peut pas modifier la propriété de la classe IOC importée. Si elle désire le faire, elle doit utiliser l'héritage pour définir une nouvelle classe qui lui est propre.

Annexe C

Répertoire UML MSIG

(Cette annexe fait partie intégrante de la présente Recommandation.)

Ci-après figurent des lignes directrices pour la spécification des résultats de la phase d'analyse reposant sur le répertoire UML (langage de modélisation unifié) du 3GPP [b-3GPP TS 32.152].

C.1 Introduction

Le langage UML fournit un vaste ensemble de concepts, de notations et d'éléments de modèle pour la modélisation des systèmes distribués. Dans le cadre des spécifications de l'analyse, il n'est pas nécessaire d'utiliser la totalité des notations et des éléments de modèle du langage. La présente annexe dresse la liste des notations et éléments de modèle UML nécessaires et suffisants, y compris ceux construits à l'aide du mécanisme d'extension UML <<stéréotype>>, aux fins de l'élaboration de spécifications indépendantes du protocole. Cet ensemble de notations et éléments de modèle est appelé le "répertoire de modélisation UML".

Les recommandations qui suivent la méthodologie doivent utiliser les notations et les éléments de modèle UML de ce répertoire; elles peuvent aussi avoir recours à d'autres notations et éléments de modèle UML qui apparaîtraient nécessaires.

C.2 Eléments de modèle de base

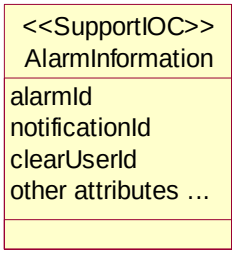
C.2.1 Généralités

Le langage UML définit un certain nombre d'éléments de modèle de base. Le présent sous-paragraphe donne le sous-ensemble des éléments à inclure dans le répertoire. La sémantique de ces éléments est définie dans [OMG UML].

C.2.2 Attribut

Voir le sous-paragraphe 3.25 de [OMG UML].

L'exemple ci-dessous montre quelques attributs, qui sont énumérés sous forme de chaînes de caractères dans le compartiment de la classe AlarmInformation réservé aux attributs.



C.2.3 Agrégation

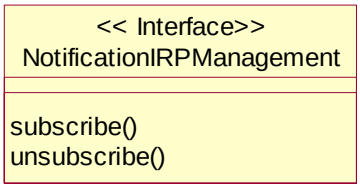
Voir le sous-paragraphe 3.43.2.5 de [OMG UML].

L'échantillon ci-dessous montre un losange blanc attaché à l'extrémité de l'arc pour indiquer une agrégation. Le losange est attaché à la classe agrégat.

C.2.4 Opération

Voir le sous-paragraphe 3.26 de [OMG UML].

L'exemple suivant montre deux opérations – matérialisées par des chaînes de caractères dans le compartiment "opérations" de la classe NotificationIRPManagement – pour lesquelles l'instance de NotificationIRPManagement peut recevoir un ordre d'exécution. L'opération possède un nom, par exemple "s'abonner" (*subscribe*), ainsi qu'une liste d'arguments (non représentée sur la figure).

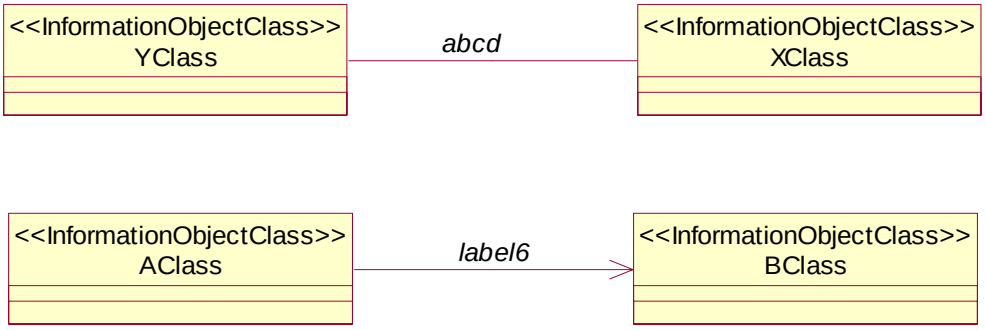


C.2.5 Association et nom d'association

Voir le sous-paragraphe 3.41 de [OMG UML].

Les deux exemples ci-dessous illustrent une association binaire entre deux éléments de modèle exactement. L'association offre la possibilité de mettre en relation un élément de modèle avec lui-même. Dans le premier exemple, l'association est de type bidirectionnel, de sorte que chaque élément de modèle a connaissance de l'autre. Dans le second exemple, l'association est de type unidirectionnel (association matérialisée par une flèche ouverte à l'extrémité de l'élément de modèle cible), de sorte que seul l'élément de modèle source a connaissance de l'élément de modèle cible (l'inverse n'étant pas vrai).

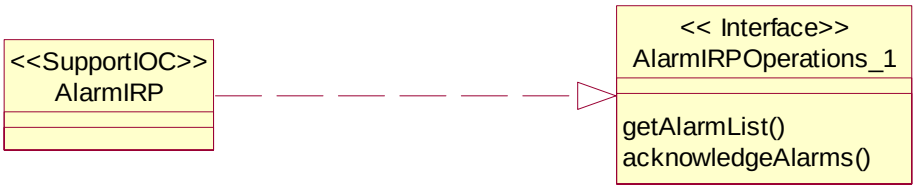
L'association peut être nommée. Par exemple *abcd* et *label6* dans les exemples suivants.



C.2.6 Relation de réalisation

Voir le sous-paragraphe 2.5.2.1 de [OMG UML].

L'exemple ci-dessous montre la relation de réalisation entre deux éléments de modèle, en l'occurrence AlarmIRPOperations_1 et AlarmIRP. Le deuxième (élément de modèle cible) implémente le premier. L'élément de modèle cible doit être une <<Interface>>.



C.2.7 Relation de généralisation

Voir le sous-paragraphe 3.50 de [OMG UML].

L'exemple ci-dessous présente une relation de généralisation entre un élément général (l'agent) et un élément spécifique (Agent_vendor_A), lequel est pleinement compatible avec le premier et vient ajouter des informations.

C.2.8 Relation de dépendance

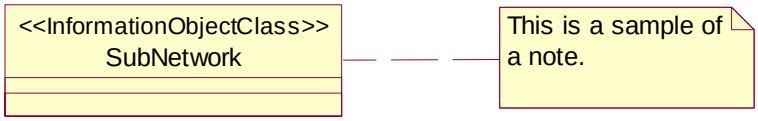
Voir le sous-paragraphe 3.51 de [OMG UML].

Dans l'exemple ci-dessous, les instances de BClass ont une relation sémantique avec les instances de AClass. Cette relation permet d'indiquer qu'une modification de l'élément cible nécessite une modification de l'élément source lié par la relation de dépendance.

C.2.9 Note

Voir le sous-paragraphe 3.11 de [OMG UML].

Cet exemple montre une note, laquelle est matérialisée par un rectangle dont le coin supérieur droit est corné. La note contient un texte; ce texte n'est pas soumis à des règles particulières. La note apparaît sur un diagramme; elle peut être reliée à un ou plusieurs éléments de modélisation par une ligne pointillée, ou ne pas être reliée.



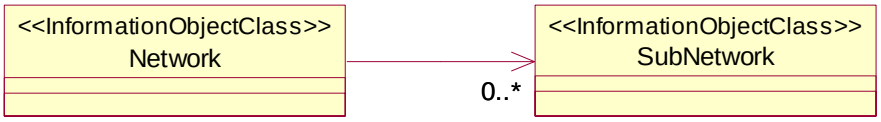
C.2.10 Multiplicité (aussi dénommée cardinalité)

Voir le sous-paragraphe 3.44 de [OMG UML].

L'exemple suivant illustre une multiplicité attachée à l'extrémité d'un arc d'association. Cette multiplicité signifie "un vers plusieurs". La ou les instances de la classe Network sont associées à zéro, une ou plusieurs instances de la classe SubNetwork.

Dans les précédentes versions de la spécification technique [b-3GPP TS 32.152], une cardinalité égale à "zéro" pouvait indiquer que la classe IOC possédait la caractéristique dite "état transitoire". Elle indiquait par exemple que l'instance n'était pas encore créée, mais qu'elle était en cours de création. Dans la présente version de la méthodologie, la cardinalité "zéro" n'est pas utilisée pour

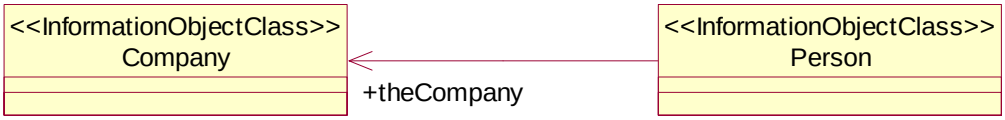
indiquer cette caractéristique, qui est considérée comme inhérente à toutes les classes IOC. On considère donc que toutes les classes IOC définies possèdent une caractéristique inhérente "état transitoire".



C.2.11 Nom de rôle

Voir le sous-paragraphe 3.43.2.6 de [OMG UML].

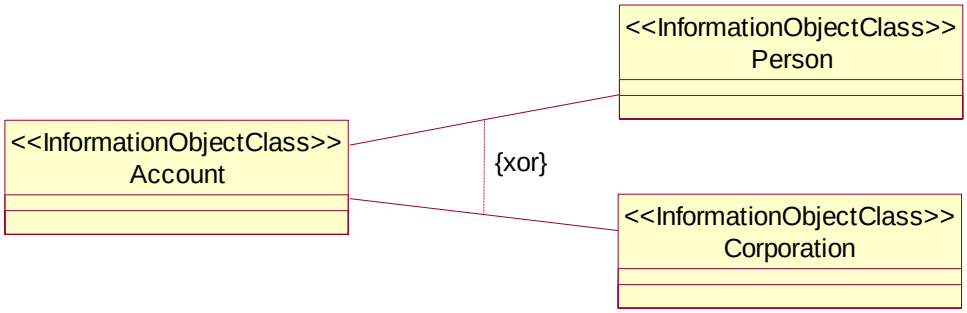
Dans l'exemple suivant, une Personne (*Person*) (instance John par exemple) est associée à une Entreprise (*Company*) (dont le DN est "Company=XYZ"). L'association est parcourue en utilisant l'extrémité opposée de l'association; ainsi, Person.theCompany de John contient le DN, à savoir "Company=XYZ". La dénomination est utilisée comme nom de rôle (*rolename*).



C.2.12 Contrainte Xor

Voir les sous-paragraphes 2.5.2.3 et 3.42.5.1 de [OMG UML].

L'exemple suivant montre un *Account* (compte) (par exemple le compte 0960) qui est associé à une *Person* (personne) (par exemple John Smith) ou à une *Corporation* (entreprise) (par exemple ABC Inc.).



C.3 Stéréotypes

C.3.1 Généralités

Le présent sous-paragraphe énumère l'ensemble des stéréotypes qu'il est autorisé d'utiliser dans des spécifications d'interfaces de gestion. On notera que l'un de ces stéréotypes, <<Interface>>, est déjà défini dans [OMG UML]. La présente Recommandation le mentionne par commodité et par souci de complétude, et en définit d'autres.

Tableau C.3-1 – Stéréotypes d'entités

Stéréotype	Classe de base	Eléments du métamodèle concernés
Interface [Interface]	Classe	
ProxyClass [Classe proxy]	Classe	
Notification [Notification]	Classe	
Archetype [Archétype]	Classificateur (sous-paragraphe 2.5.2.10 de [OMG UML])	
InformationObjectClass [Classe d'objets informationnels]	Classificateur	
SupportIOC [IOC de prise en charge]	Classificateur	
use (utilise)	Association	
may use (peut utiliser)	Association	
may realize (peut réaliser)	Association	
names (nomme)	Composition	

C.3.2 <<Interface>>

Sous-paragraphe 2.5.2.25 de [OMG UML]:

"Une interface est un ensemble nommé d'opérations qui caractérisent le comportement d'un élément. Dans le métamodèle, une Interface contient un ensemble d'Opérations qui, ensemble, définissent un service offert par un Classificateur réalisant l'Interface. Un Classificateur peut offrir plusieurs services – il peut donc éventuellement réaliser plusieurs Interfaces – et plusieurs Classificateurs peuvent réaliser la même Interface.

Les Interfaces peuvent [avoir ou] ne pas avoir d'Attributs, d'Associations ou de Méthodes. Une Interface peut participer à une Association à condition qu'elle ne puisse pas la voir; autrement dit, un Classificateur (autre qu'une Interface) peut avoir une Association vers une Interface, ladite association étant navigable à partir du Classificateur mais pas à partir de l'Interface."

Extrait du sous-paragraphe 2.5.4.6 de [OMG UML]:

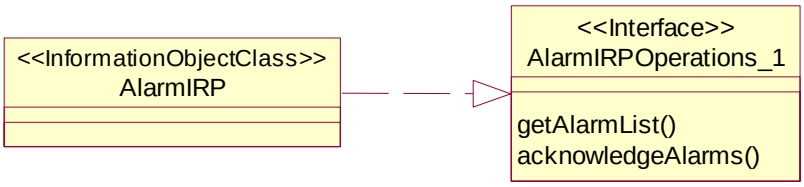
"La finalité d'une interface est de rassembler un ensemble d'opérations qui constituent un service cohérent offert par des classificateurs. Les interfaces offrent un moyen de partitionner et de caractériser des groupes d'opérations. Une interface n'est qu'un ensemble d'opérations qui porte un nom. Elle ne peut pas être directement instanciée."

Extrait du sous-paragraphe 2.5.4.6 de [OMG UML]:

"Plusieurs classificateurs peuvent réaliser la même interface. Tous doivent contenir au minimum les opérations qui correspondent à celles contenues dans l'interface. La spécification d'une opération contient la signature de l'opération (à savoir, son nom, les types des paramètres et le type du résultat de l'opération). Une interface ne présuppose pas de la structure interne du classificateur de réalisation. Par exemple, elle ne précise pas quel algorithme il convient d'utiliser pour réaliser une opération. Une opération peut néanmoins inclure une spécification des effets qui résultent de son appel [par exemple, à l'aide de pré- et post-conditions]."

C.3.2.1 Exemple

L'exemple suivant montre une <<Interface>> appelée AlarmIRPOperations_1 qui possède deux opérations. Les paramètres d'entrée et de sortie des opérations sont cachés (c'est-à-dire qu'ils ne sont pas montrés). AlarmIRP a une relation de réalisation obligatoire unidirectionnelle avec l'<<Interface>>.



Notation adoptée pour une <<Interface>>

C.3.3 <<ProxyClass>>

C.3.3.1 Généralités

Le stéréotype <<ProxyClass>> représente un certain nombre de <<InformationObjectClass>>. Il encapsule les attributs, les liens, les méthodes (opérations) et les interactions qui sont présents dans l'élément <<InformationObjectClass>> représenté.

La sémantique du stéréotype <<ProxyClass>> est que tous les comportements de <<ProxyClass>> sont présents dans l'élément <<InformationObjectClass>> représenté. Etant donné que cette classe est simplement une représentation d'autres classes, elle ne peut pas définir son propre comportement et adopte les comportements qui sont déjà définis par l'élément <<InformationObjectClass>> représenté.

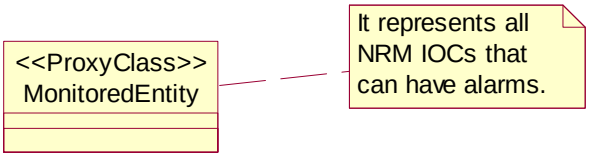
Un <<InformationObjectClass>> donné peut être représenté par zéro, un ou plusieurs stéréotypes <<ProxyClass>> ou <<Archétype>>. Par exemple, l'<<InformationObjectClass>> ManagedElement (élément géré) peut avoir un stéréotype <<ProxyClass>> MonitoredEntity (entité surveillée) et un stéréotype <<ProxyClass>> ManagedEntity (entité gérée).

L'entité source qui a une association avec l'entité <<ProxyClass>> peut accéder aux attributs de cette dernière.

C.3.3.2 Exemple

L'exemple suivant illustre l'entité <<ProxyClass>> appelée MonitoredEntity (entité surveillée). Elle représente toutes les classes <<InformationObjectClass>> du modèle de ressource de réseau (par exemple, la classe <<InformationObjectClass>> GgsnFunction) dont les instances font l'objet d'une surveillance des conditions d'alarme.

A noter que <<MonitoredEntity>> ne définit aucun attribut. Les attributs sont en effet déjà définis par toutes les <<InformationObjectClass>> représentées par <<MonitoredEntity>>.



Notation adoptée pour une <<ProxyClass>>

On trouvera à l'Appendice V d'autres exemples utilisant les <<ProxyClass>>.

C.3.4 <<Archétype>>

C.3.4.1 Généralités

Le stéréotype <<Archétype>> représente un ensemble de propriétés communes de classes (par exemple, des attributs, des liens, des opérations et des interactions que l'on trouve ordinairement associés à l'élément <<InformationObjectClass>> représenté).

La sémantique de l'<<Achétype>> est la suivante: tous les attributs, liens, opérations et interactions encapsulés par l'<<Archétype>> sont éventuellement présents dans l'élément <<InformationObjectClass>> représenté. L'<<Archétype>> constitue une classe "emplacement" (*placeholder*) des plus utiles dans les modèles d'analyse indépendants de la technologie dont la spécification sera ultérieurement complétée et/ou qui devront être mis en correspondance avec un modèle de construction plus complet.

C.3.4.2 Exemple

L'exemple suivant illustre un <<Archétype>> appelé StateManagement (gestion d'état) ainsi qu'un Agent <<InformationObjectClass>> qui dépend de ce StateManagement. A noter que pour le StateManagement, plusieurs attributs ont été définis (non indiqués dans le diagramme UML). Les classes qui dépendent de ce StateManagement peuvent éventuellement utiliser tous les attributs du StateManagement, au moins un des attributs de StateManagement étant présent dans l'Agent. La liste précise des attributs du StateManagement utilisés par l'Agent est indiquée dans la spécification de l'Agent.

Notation adoptée pour <<Archetype>>

C.3.5 <<InformationObjectClass>>

C.3.5.1 Généralités

Le stéréotype <<InformationObjectClass>> représente une classe d'objets informationnels (IOC). Chaque <<InformationObjectClass>> représente un ensemble d'instances de mêmes structures, comportements et relations.

La classe <<InformationObjectClass>> ainsi que d'autres classes d'information telles que <<Interface>> sont mises en correspondance avec des éléments du modèle dépendant de la technologie, par exemple la classe d'objets gérés (*Managed Object Class*) définie dans les lignes directrices GDMO pour le protocole CMIP. La mise en correspondance des éléments de modélisation indépendants du protocole avec des éléments de modélisation dépendants du protocole est documentée dans les spécifications dépendantes du protocole correspondantes.

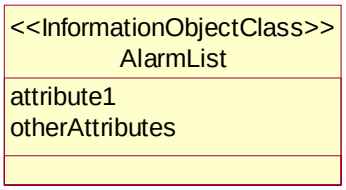
La portée du nom d'une classe <<InformationObjectClass>> est celle de la Recommandation dans laquelle la classe est spécifiée; ce nom ne doit pas être réutilisé pour d'autres classes <<InformationObjectClass>> de la même Recommandation (unicité du nom). Le nom de la Recommandation est traité de manière analogue au nom d'un Package UML.

La classe <<InformationObjectClass>> est identique à une classe UML si ce n'est qu'elle n'inclut pas/ne définit pas de méthodes ni d'opérations.

Sous-paragraphe 3.22.1 de [OMG UML]: "Une classe représente un concept au sein du système en cours de modélisation. Les classes possèdent une structure de données et un comportement, ainsi que des relations avec d'autres éléments."

C.3.5.2 Exemple

L'exemple suivant illustre une classe <<InformationObjectClass>> appelée AlarmList (liste des alarmes).



Notation adoptée pour <<InformationObjectClass>>

C.3.6 <<utilise>> et <<peut utiliser>>

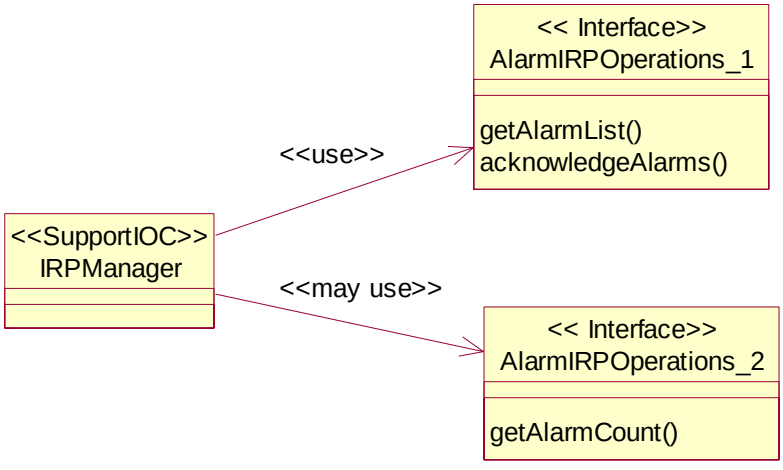
Les stéréotypes <<utilise>> (<<use>>) et <<peut utiliser>> (<<may use>>) désignent des associations unidirectionnelles. La cible doit être une <<Interface>> ou une <<Notification>>.

Lorsque la cible est une <<Interface>>, l'association <<utilise>> indique que la classe source doit être en mesure d'utiliser l'<<Interface>> cible, c'est-à-dire d'invoquer les opérations définies par cette <<Interface>>. Cette possibilité offerte par l'entité source doit obligatoirement être prise en charge. L'association <<peut utiliser>> indique que la classe source peut éventuellement avoir la possibilité d'utiliser l'<<Interface>> cible, c'est-à-dire d'invoquer les opérations définies par cette <<Interface>>. Dans le cas du stéréotype <<peut utiliser>>, il n'est pas obligatoire de prendre en charge cette possibilité offerte par l'entité source.

Lorsque la cible est une <<Notification>>, l'association <<utilise>> indique que la classe source doit être l'initiatrice des notifications définies par la <<Notification>> cible. Cette capacité de l'entité source doit obligatoirement être prise en charge. L'association <<peut utiliser>> indique que la classe source peut éventuellement être l'initiatrice des notifications définies par la <<Notification>> cible. Dans le cas du stéréotype <<peut utiliser>>, il n'est pas obligatoire de prendre en charge cette possibilité offerte par l'entité source.

C.3.6.1 Exemple d'une <<Interface>> cible

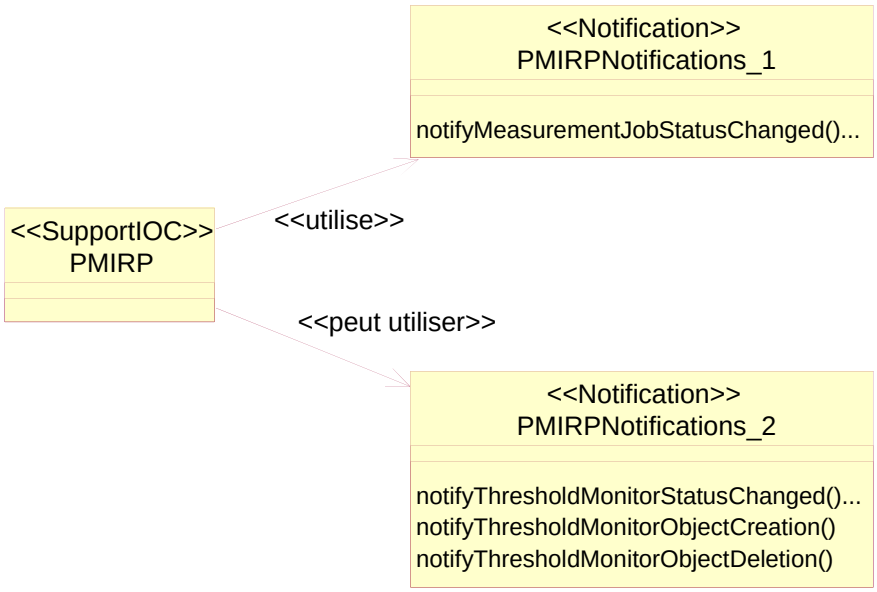
Dans le diagramme ci-dessous, l'IRPManager (gestionnaire de point IRP) doit utiliser les opérations définies par AlarmIRPOperations_1 et peut éventuellement utiliser les opérations définies par AlarmIRPOperations_2.



**Notations adoptées pour <<utilise>> et <<peut utiliser>>
dans le cas où la cible est une <<Interface>>**

C.3.6.2 Exemple d'une <<Notification>> cible

Dans le diagramme ci-dessous, le PMIRP doit être en mesure d'émettre les notifications définies par PMIRPNotifications_1 ou d'en être à l'origine, et peut avoir la possibilité d'émettre les notifications définies par PMIRPNotifications_2 ou d'en être à l'origine.



Notations adoptées pour <<utilise>> et <<peut utiliser>>
dans le cas où la cible est une <<Notification>>

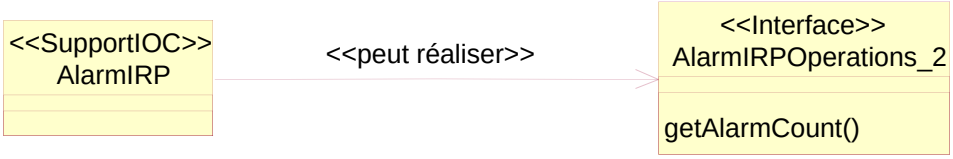
C.3.7 <<peut réaliser>>

Le stéréotype <<peut réaliser>> désigne une association unidirectionnelle. La cible doit être une <<Interface>>. L'association <<peut réaliser>> indique que l'entité source peut éventuellement réaliser les opérations définies par l'<<Interface>> cible.

A noter que l'association "réalise" figure parmi les éléments de base UML (et qu'il n'est donc pas nécessaire de définir un stéréotype pour ce type d'association). L'association <<réalise>> indique que l'entité source doit réaliser (ou implémenter) les opérations définies par l'<<Interface>> cible.

C.3.7.1 Exemple

Dans le diagramme ci-dessous, AlarmIRP peut éventuellement réaliser l'opération d'AlarmIRPOperations_2.



Notation adoptée pour <<peut réaliser>>

C.3.8 <<nomme>>

Le stéréotype <<nomme>> spécifie une composition unidirectionnelle. L'instance cible est identifiable de façon unique, au sein de l'espace de nommage de l'entité source, parmi toutes les autres instances cibles du même classificateur cible et parmi les autres instances cibles d'autres classificateurs ayant avec la source la même relation de composition <<nomme>>.

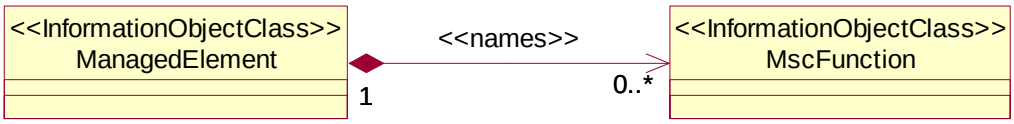
Le classificateur source et le classificateur cible doivent tous deux avoir un attribut de dénomination.

Le fait d'utiliser la composition comme l'acte de conteneur de nom fournit une sémantique du type "relation Tout-Partie" entre le domaine et les éléments nommés qui sont contenus, ne serait-ce que par leur nom. Du point de vue de la gestion, l'accès à la partie se fait par le tout. La multiplicité doit être indiquée aux deux extrémités de la relation.

Une instance cible ne peut pas avoir plusieurs relations <<nomme>> avec plusieurs sources; autrement dit, une instance cible ne peut pas participer ni appartenir à plusieurs espaces de nommage.

C.3.8.1 Exemple

Dans le diagramme ci-dessous, les instances de MscFunction sont toutes identifiables de façon unique au sein de l'espace de nommage d'une instance de ManagedElement.



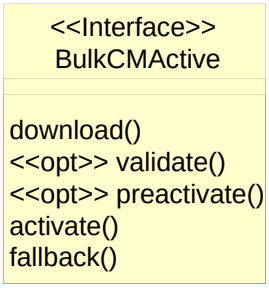
Notation adoptée pour <<nomme>>

C.3.9 <<opt>>

La notation <<opt>> (ou <<optionnel>>) permet d'indiquer le caractère optionnel des attributs, des paramètres et des opérations (respectivement) dans les diagrammes UML.

En l'absence de ce stéréotype, l'attribut, le paramètre ou l'opération en question est obligatoire.

C.3.9.1 Exemple



Notation <<opt>> appliquée à des opérations

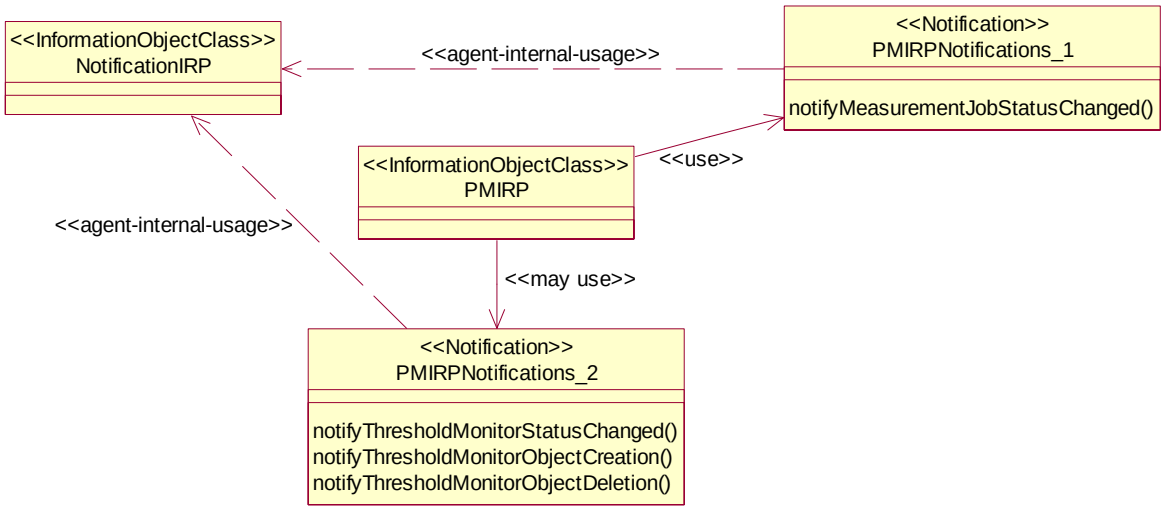
C.3.10 <<Notification>>

C.3.10.1 Généralités

Le stéréotype <<Notification>> désigne un ensemble nommé de notifications.

C.3.10.2 Exemple

L'exemple ci-dessous présente deux <<Notification>>, l'une nommée "PMIRPNotifications_1", l'autre "PMIRPNotifications_2". Toutes les deux possèdent une ou plusieurs notifications. Exemple de notification: notifyMeasurementJobStatusChanged().



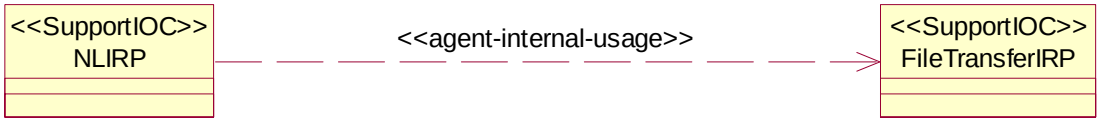
Notation adoptée pour <<Notification>>

C.3.11 <<utilisation-interne-agent>>

Le stéréotype <<utilisation-interne-agent>> désigne une association unidirectionnelle. La source transmet des informations de gestion de réseau à la cible. La source et la cible sont des entités ou des processus qui s'exécutent dans des instances de point IRP différentes, par exemple AlarmIRP ou PMIRP. Les instances peuvent être contenues via leur nom dans les mêmes instances ou dans des instances différentes d'IRPAgent. Les informations de gestion de réseau transmises et le mécanisme de transfert de ces informations ne sont pas normalisés; ils sont propres au fabricant.

C.3.11.1 Exemple

Dans le diagramme ci-dessous, NLIRP (point IRP NotificationLog) peut transmettre des informations de gestion de réseau à FTIRP (FileTransferIRP).



Notation adoptée pour <<utilisation-interne-agent>>

C.3.12 <<SupportIOC>>

Le stéréotype <<SupportIOC>> désigne le descripteur d'un ensemble de possibilités de gestion.

<<SupportIOC>> est identique à une *classe* UML si ce n'est qu'il n'inclut pas/ne définit pas de méthodes ni d'opérations.

Sous-paragraphe 3.22.1 de [OMG UML]: "Une *classe* représente un concept au sein du système en cours de modélisation. Les classes possèdent une structure de données et un comportement, ainsi que des relations avec d'autres éléments."

C.3.12.1 Exemple

L'exemple suivant illustre une <<SupportIOC>> appelée AlarmList (liste des alarmes).

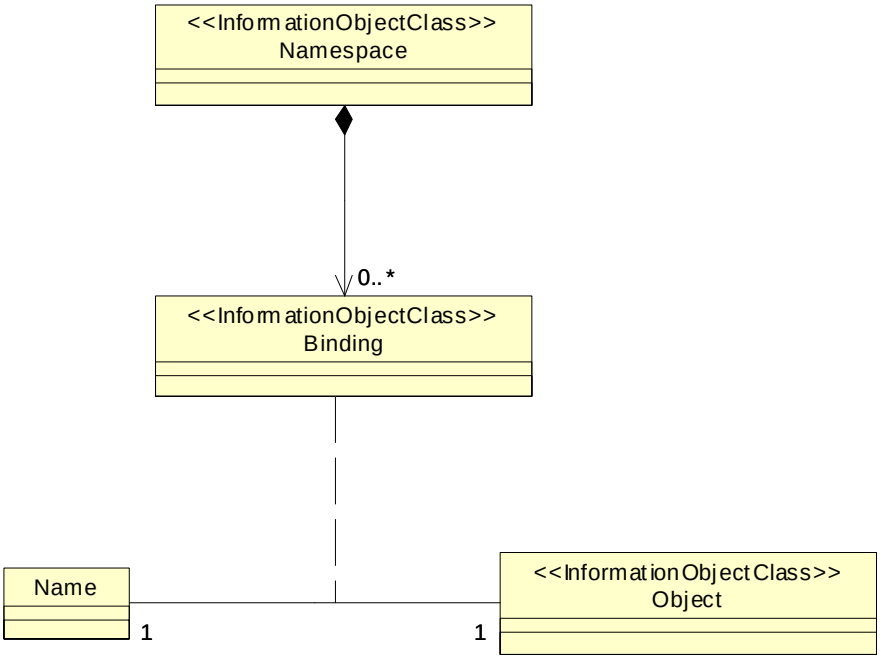
Notation adoptée pour <<SupportIOC>>

C.4 Classes d'associations

Le sous-paragraphe 3.46 de [OMG UML] définit une classe d'associations comme suit:

> "Une classe d'associations est une association qui possède aussi des propriétés de classe (ou une classe qui possède des propriétés d'association). Bien qu'elle soit graphiquement représentée comme une association et une classe, elle n'est en fait qu'un seul et même élément de modèle".

Les classes d'associations sont une bonne solution à utiliser lorsqu'une "InformationObjectClass" (classe d'objets informationnels) doit maintenir des associations avec plusieurs autres classes d'objets informationnels et qu'il existe des relations entre les membres des associations dans le périmètre de la classe "InformationObjectClass" "contenante". Par exemple, un espace de nommage (*namespace*) maintient un ensemble de rattachements (*bindings*), un rattachement reliant un nom à un objet. Une "IOC" de rattachement peut être modélisée sous la forme d'une classe d'associations qui fournit la sémantique de rattachement à la relation entre un nom et d'autres "InformationObjectClass". Ce cas est illustré à la figure suivante (à titre d'exemple seulement; non repris d'une autre Recommandation).



Exemple de classe d'associations

C.5 Classe abstraite

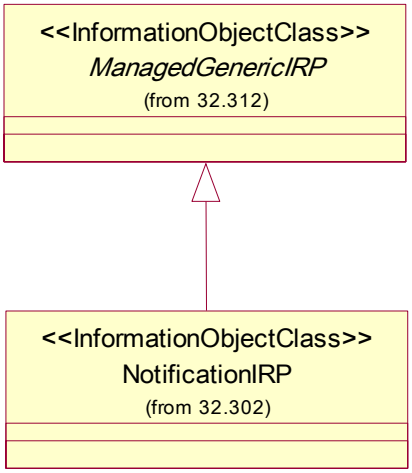
C.5.1 Généralités

Une classe abstraite désigne une <<InformationObjectClass>> comme une classe de base qui sera héritée par des sous-classes. Une classe abstraite ne peut pas être instanciée.

Notation adoptée pour une classe abstraite: le nom de classe de la <<InformationObjectClass>> correspondante est indiqué en italique dans le diagramme.

C.5.2 Exemple

Le diagramme suivant montre que ManagedGenericIRP est une classe <<InformationObjectClass>> abstraite.



Notation adoptée pour les classes abstraites

C.6 Application de <<InformationObjectClass>> et <SupportIOC>>

<<InformationObjectClass>> et <SupportIOC>> sont des stéréotypes. Ces deux stéréotypes servent un objectif analogue en cela que chacun est un ensemble nommé de propriétés de gestion. Cela étant, leurs applications, dans le contexte de la prise en charge de la gestion via une interface de gestion, peuvent être différentes. Le présent paragraphe souligne les similitudes et les différences de ces applications.

	<<InformationObjectClass>>	**<<SupportIOC>>**
Peut être une classe abstraite?	Oui	Oui
Peut être une classe concrète?	Oui	Oui
Peut hériter de <<InformationObjectClass>>?	Oui	Non
Peut hériter de <<SupportIOC>>?	Non	Oui
Peut être contenu, via le nom, par <<InformationObjectClass>>?	Oui	Oui
Peut être contenu, via le nom, par <<SupportIOC>>?	Non	Oui

	<<InformationObjectClass>>	**<<SupportIOC>>**
Une instance peut-elle avoir un DN?	<<InformationObjectClass>> doit être une classe d'un arbre de nommage, ce qui signifie que toutes ses instances doivent avoir un DN.	<<SupportIOC>> peut être utilisé par l'auteur des spécifications pour une classe à l'intérieur d'un arbre de nommage. En pareil cas, cela signifie que toutes ses instances auront un DN.
Un Gestionnaire peut-il recevoir des informations par le biais de notifications dont les paramètres objectClass et objectInstance contiennent le DN de l'instance?	Oui. Les types de notification émis figurent dans le tableau des notifications associé à la définition de classe.	Oui si <<SupportIOC>> est une classe d'un arbre de nommage. Les types de notification émis figurent dans le tableau des notifications associé à la définition de classe. Non si <<SupportIOC>> n'est pas une classe d'un arbre de nommage.

Annexe D

Conception

(Cette annexe fait partie intégrante de la présente Recommandation.)

La présente annexe fournit des lignes directrices pour la spécification des conceptions dépendantes du protocole. Elle fera l'objet d'une étude ultérieure.

Annexe E

Définitions des types d'information – répertoire des types

(Cette annexe fait partie intégrante de la présente Recommandation.)

La présente annexe définit un répertoire de types à utiliser pour spécifier les informations de type dans le modèle conceptuel (modèle d'analyse/service d'information).

Le répertoire est constitué d'un sous-ensemble des types définis dans ASN.1 [UIT-T X.680] et d'autres types, qui ont été élaborés à partir des types définis dans ASN.1 (§ E.4).

Les mots clés à utiliser pour chaque type sont récapitulés dans le Tableau E.1.

E.1 Types de base

Les types de base sont des types directement utilisables pour définir des attributs et des paramètres. Ils peuvent aussi servir à construire des types complexes. Les types de base comprennent les types ASN.1 suivants:

E.1.1 Type entier, § 19 de [UIT-T X.680]

E.1.2 Type réel, § 21 de [UIT-T X.680]

E.1.3 Type booléen, § 18 de [UIT-T X.680]

E.1.4 Type chaîne binaire, § 22 de [UIT-T X.680]

E.1.5 Type néant, § 24 de [UIT-T X.680]

E.1.6 Type temps généralisé, § 38 de [UIT-T X.680]

E.2 Type énuméré

Le type énuméré, § 20 de [UIT-T X.680], représente une énumération de valeurs. Toutes les valeurs que peut prendre un attribut ou un paramètre donné doivent être énumérées dans les colonnes correspondant aux valeurs autorisées de l'attribut ou du paramètre. Seul le style "noms énumérés" est applicable au modèle conceptuel, autrement dit, l'identification des valeurs concrètes (nombres ou chaînes) est laissée à la discrétion des modèles de conception concrète.

NOTE – Si le nombre de ces valeurs est supérieur à 50, il est recommandé de les définir dans un appendice ou un document séparé.

E.3 Types complexes

Les types complexes peuvent être définis à partir des concepts suivants:

E.3.1 Types séquence, § 25 de [UIT-T X.680]

E.3.2 Types choix, § 29 de [UIT-T X.680]

E.3.3 Types ensemble, § 27 de [UIT-T X.680]

De plus, les listes et les ensembles de types complexes sont pris en charge à l'aide des types suivants:

E.3.4 Types séquence-de, § 26 de [UIT-T X.680]

E.3.5 Types ensemble-de, § 28 de [UIT-T X.680]

E.4 Types utiles

E.4.1 Type chaîne

Une "chaîne" représente une chaîne de caractères, l'ensemble des caractères n'étant pas limité, soit:

Chaîne::= **UnrestrictedCharacterStringType,** § 44 de [UIT-T X.680]

E.4.2 Type nom

Un "nom" représente un nom exclusif d'une instance d'objet dans un espace de nommage. Il peut inclure des informations sur la hiérarchie de l'arbre de contenance d'objet, mais il est dépendant de l'implémentation et sort du champ d'application de la présente Recommandation. Sur le plan formel, le type nom est défini comme suit:

Nom::= TYPE-IDENTIFIER, Annexe A de [UIT-T X.681]

E.5 Mots clés

Le Tableau E.1 dresse la liste des mots clés à utiliser dans le modèle d'analyse (voir l'Annexe B) pour la définition du type d'information, par exemple:

Nom du paramètre	**Qualificatif de prise en charge (*Support qualifier*)**	**Type d'information/ Valeurs autorisées**	**Remarque**
…			
eventIdList	M	ENSEMBLE D'ENTIERS/–	Liste des alarmes dont il faut accuser réception.

Tableau E.1 – Mots clés

Type	**Mot clé**
Type entier	INTEGER
Type réel	REAL
Type booléen	BOOLEAN
Type chaîne binaire	BIT STRING
Type néant	NULL
Type temps généralisé	GeneralizedTime
Type énuméré	ENUMERATED
Type séquence	SEQUENCE
Type choix	CHOICE
Type ensemble	SET
Type séquence-de	SEQUENCE OF
Type ensemble-de	SET OF
Type chaîne	String
Type nom	Name

Annexe F

Lignes directrices relatives aux propriétés des classes IOC, à l'héritage et à l'importation d'entités

(Cette annexe fait partie intégrante de la présente Recommandation.)

Les présentes lignes directrices ont été établies à partir de [b-3GPP TS 32.150].

F.1 Propriété de classe IOC

Les propriétés des classes IOC (y compris des classes IOC de prise en charge (*Support IOC*)) sont spécifiées sur la base des informations suivantes:

a) Un ou plusieurs attributs de la classe IOC, y compris sa sémantique et sa syntaxe, ses intervalles de valeurs autorisées et ses qualificatifs de prise en charge. Les attributs de classe IOC ne se limitent pas à la gestion de la configuration; ils incluent aussi ceux concernant, par exemple, 1) la gestion de la qualité de fonctionnement (à savoir, les types de mesure), 2) la gestion des traces et 3) la gestion de la comptabilisation.

b) Le comportement non spécifique à un attribut associé à une classe IOC.

NOTE 1 – Par exemple, le lien entre MscServerFunction et CsMgwFunction est facultatif. Il est obligatoire si l'instance MscServerFunction appartient à une instance de ManagedElement et que l'instance CsMgwFunction appartient à une autre instance de ManagedElement. Ce comportement "lien" est un comportement non spécifique à un attribut. En principe, ce comportement, comme d'autres, sera hérité.

c) Une ou plusieurs relations de la classe IOC avec une ou plusieurs autres IOC.

d) Un ou plusieurs types de notification de la classe IOC et leurs qualificatifs.

e) La relation d'une classe IOC avec ses parents (voir la Note 2). On dénombre trois cas qui s'excluent mutuellement:

1) La classe IOC peut avoir n'importe quel parent. Dans le diagramme UML, la classe a pour parent Any.

2) La classe IOC est abstraite, tous les parents possibles (un seul éventuellement) ont été désignés, et les sous-classes IOC peuvent être désignées comme une classe IOC racine. Dans un diagramme UML, la classe a zéro ou plusieurs parents possibles de classes spécifiques (sauf Any).

3) La classe IOC est concrète, tous les parents possibles (un seul éventuellement) ont été désignés, et la classe IOC peut être désignée comme une classe IOC racine. Dans un diagramme UML, la classe a un ou plusieurs parents possibles de classes spécifiques (sauf Any).

De deux choses l'une: soit l'instance d'une classe IOC est une classe IOC racine, soit elle possède un et un seul parent. Seul le 3GPP SA5 peut désigner une classe IOC comme une IOC racine potentielle. Actuellement, seules les classes IOC SubNetwork, ManagedElement et MeContext peuvent être des IOC racines.

NOTE 2 – La relation parent-enfant dans ce sous-paragraphe correspond à la relation "nom du parent-contient l'enfant".

f) La relation d'une classe IOC avec ses enfants. On dénombre trois cas qui s'excluent mutuellement:

1) Une classe IOC ne peut pas avoir de classes IOC enfants (relation nom-conteneur). Dans le diagramme UML, la classe n'a pas d'enfant.

2) Une classe IOC peut avoir une ou plusieurs classes IOC enfants. Le nombre maximal d'instances par classe IOC enfant peut être spécifié. Une classe IOC peut indiquer que les objets propres à un fabricant ne peuvent pas être des classes IOC enfants. Dans le diagramme UML, la classe a pour enfant Any.

3) Une classe IOC ne peut avoir que des classes IOC enfants spécifiques (ou leurs sous-classes). Le nombre maximal d'instances par classe IOC enfant peut être spécifié. Une classe IOC peut indiquer que les objets propres à un fabricant ne peuvent pas être des classes IOC enfants. Dans un diagramme UML, la classe a un ou plusieurs enfants de classes spécifiques (sauf Any).

g) Information indiquant si une classe IOC peut être instanciée ou non (autrement dit, si une classe IOC est une classe abstraite).

h) Un attribut à des fins de dénomination.

F.2 Héritage

Une classe IOC (la sous-classe) qui hérite d'une autre classe IOC (la superclasse) possède toutes les propriétés de la superclasse.

La sous-classe peut modifier le ou les qualificatifs de prise en charge hérités qui sont optionnels en vue de les rendre obligatoires, mais pas l'inverse. La sous-classe peut modifier le qualificatif de prise en charge hérité qui est conditionnel-optionnel en vue de le rendre conditionnel-obligatoire, mais pas l'inverse.

Une classe IOC peut être la superclasse de nombreuses classes IOC. Une sous-classe ne peut pas avoir plus d'une superclasse.

La sous-classe peut:

a) Ajouter des attributs uniques (par rapport à ceux de sa superclasse), y compris leur comportement, leurs intervalles de valeurs autorisées et leurs qualificatifs de prise en charge. Chaque attribut supplémentaire doit posséder un nom unique (parmi tous les attributs ajoutés et hérités).

b) Ajouter un comportement non spécifique à un attribut sur la base d'une classe IOC. Ce comportement ne doit pas entrer en contradiction avec le comportement de la superclasse héritée.

c) Ajouter une ou plusieurs relations avec une ou plusieurs classes IOC. Chaque relation supplémentaire doit posséder un nom unique (parmi toutes les relations ajoutées et héritées).

d) Ajouter des types de notification supplémentaires et leurs qualificatifs.

e) Désigner tous les parents possibles (éventuellement un seul) (ainsi que leurs sous-classes) si la superclasse possède la propriété-e-1, de sorte qu'une classe IOC possédera la propriété-e-2 ou la propriété-e-3. Restreindre le(s) parent(s) possible(s) (et leurs sous-classes) et/ou supprimer la possibilité pour la sous-classe d'être une classe IOC racine si la superclasse possède la proriété-e-2 ou la propriété-e-3.

f) Ajouter une ou plusieurs classes IOC enfants si la superclasse possède la propriété-f-2 de sorte qu'une classe IOC possédera la propriété-f-3. Restreindre les classes IOC enfants autorisées (ou leurs sous-classes) si la superclasse possède la propriété-f-3.

g) Préciser si une classe IOC peut être instanciée ou non (autrement dit, si la classe IOC est une classe abstraite).

h) Limiter l'intervalle des valeurs autorisées d'un attribut d'une superclasse possédant un intervalle de valeurs autorisées.

F.3 Importation d'entités (interface, IOC et attribut)

Les spécifications d'interface de gestion définissent des entités (par exemple, des classes IOC, des interfaces et des attributs). Pour faciliter la réutilisation des définitions d'entité entre spécifications d'interfaces, on a recours à un mécanisme d'importation. Lorsqu'une spécification d'interface de gestion (la spécification sujet) importe une entité définie dans une autre spécification d'interface de gestion, on considère que la spécification sujet a défini l'entité importée. De plus, la spécification sujet ne peut pas modifier les propriétés de l'entité importée. Si elle a besoin d'une entité qui n'est pas en tous points identique à l'entité importée, elle doit en définir une nouvelle, qui hérite de l'entité importée, et modifier la définition de la nouvelle entité.

Appendice I

Exemple de définition des besoins

(Cet appendice ne fait pas partie intégrante de la présente Recommandation.)

NOTE – L'exemple suivant n'est donné qu'à des fins d'illustration et n'a pas vocation à définir un ensemble complet ou correct de besoins pour la gestion des alarmes.

1 Concepts et contexte

Toute évaluation de l'état de santé des éléments de réseau et du réseau dans son ensemble intègre nécessairement la détection des pannes et donc le signalement des alarmes à l'OS (EM et/ou NM).

2 Besoins de niveau métier

2.1 Besoins

Toute panne survenant dans le réseau peut être classée dans l'une des deux catégories suivantes:

– Pannes matérielles, c'est-à-dire le dysfonctionnement d'une ressource physique à l'intérieur d'un élément de réseau.

– Problèmes logiciels: bugs informatiques, incohérence d'une base de données, etc.

2.1.1 Détection des pannes

REQ-FM-FON-01 Il faudra définir, pour la plupart des pannes, des contions permettant de statuer sur leur présence ou leur absence, autrement dit, des conditions de survenue de panne et de disparition de panne. Tout incident de ce genre doit être mentionné dans le présent appendice comme une panne ADAC. Les entités de réseau doivent être en mesure de savoir quand une panne ADAC détectée préalablement n'est plus présente (autrement dit, l'annulation de la panne), en ayant recours à des techniques analogues à celles qu'elles utilisent pour détecter sa survenue.

2.1.2 Annulation des alarmes

Les alarmes déclenchées à la suite d'une panne doivent être annulées. Pour annuler une alarme, il est en général nécessaire de réparer la panne correspondante.

…

REQ-FM-FON-02 Chaque fois qu'une alarme est annulée, l'Agent doit générer un événement d'annulation d'alarme. L'annulation d'alarme est définie comme une alarme.

2.1.3 Transmission et filtrage des alarmes

REQ-FM-FON-03 Pour chaque panne détectée, des alarmes appropriées (notifications de la panne) doivent être générées par l'entité de réseau défectueuse.

…

2.2 Rôles des acteurs

Système géré L'entité jouant un rôle d'agent.

Système de gestion L'entité jouant le rôle de gestionnaire.

2.3 Ressources de télécommunication

L'équipement de réseau géré est vu comme un ensemble de ressources de télécommunication pertinentes dans la présente Recommandation.

2.4 Diagrammes de cas d'utilisation de haut niveau

2.4.1 Signaler une alarme

Le premier diagramme de cas d'utilisation représenté à la Figure I.1 illustre l'interaction globale de l'interface de gestion des alarmes.

Il montre les interactions qui interviennent lorsque la détection d'une panne est signalée.

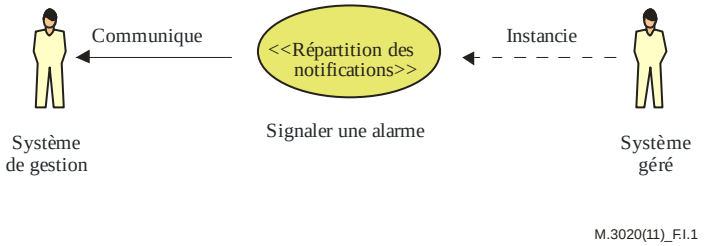


Figure I.1 – Signaler une alarme

3 Besoins de niveau spécification

3.1 Besoins

Il n'y a pas de besoins de niveau spécification.

3.2 Rôles des acteurs

Voir le sous-paragraphe 2.2 du présent modèle.

3.3 Ressources de télécommunication

Voir le sous-paragraphe 2.3 du présent modèle.

3.4 Cas d'utilisation

3.4.1 Signaler la panne

Etape du cas d'utilisation	Evolution/Spécification	Utilisation de la relation <<utilise>>
Objectif (*)	Lorsqu'une panne est détectée, le système géré envoie au système de gestion une notification de rapport d'alarme du type correspondant, via l'interface Q.	
Acteurs et rôles (*)	Le système de gestion est un consommateur de notifications émises par le système géré.	
Ressources de télécommunication	Toute entité gérée.	
Hypothèses	Une panne est détectée.	
Préconditions	Il existe un canal de communication ouvert entre le système de gestion et le système géré.	
Commence lorsque	Une panne est détectée.	
Etape 1 (*)	Lorsqu'une panne est détectée, un rapport d'alarme approprié ou un rapport d'alarme de sécurité est créé.	
Se termine lorsque	Un rapport d'alarme ou un rapport d'alarme de sécurité est envoyé par l'agent.	

Etape du cas d'utilisation	Evolution/Spécification	Utilisation de la relation <<utilise>>
Exceptions	Un dysfonctionnement dans la communication ou le processus pourrait empêcher l'envoi du rapport d'alarme au système de gestion. Le cas d'utilisation relatif à la synchronisation des alarmes couvre cette situation.	
Postconditions	Le système de gestion est informé de la panne survenue dans le système géré.	
Traçabilité (*)	REQ-FM-FON-01, REQ-FM-FON-02, etc.	

3.4.2 Annuler l'alarme

…

3.4.3 Accuser réception de l'alarme

…

Appendice II

Exemple d'analyse

(Cet appendice ne fait pas partie intégrante de la présente Recommandation.)

NOTE – L'exemple suivant, qui concerne la gestion des alarmes, n'est donné qu'à des fins d'illustration et n'a pas vocation à définir un ensemble complet ou correct de besoins pour la gestion des alarmes.

1 Concepts et contexte

Toute évaluation de l'état de santé des éléments de réseau et du réseau dans son ensemble intègre nécessairement la détection des pannes et donc le signalement des alarmes à l'OS (EM et/ou NM).

…

2 Classe d'objets informationnels

2.1 Entités d'information importées et étiquettes locales

Référence de l'étiquette	Etiquette locale
3GPP TS 32.302, classe d'objets informationnels, NotificationIRP	NotificationIRP
3GPP TS 32.302, interface, notificationIRPNotification	NotificationIRPNotification
3GPP TS 32.302, classe d'objets informationnels, IRPAgent	IRPAgent
3GPP TS 32.302, classe d'objets informationnels, ManagedGenericIRP	ManagedGenericIRP

2.2 Diagramme de classes

Ce sous-paragraphe présente l'ensemble des classes d'objets informationnels (IOC) qui encapsulent les informations au sein de l'agent. L'objectif est de recenser les informations nécessaires à l'implémentation des opérations et de l'émission des notifications par l'agent AlarmAgent. Le présent sous-paragraphe donne un aperçu général de toutes les classes d'objets supports en langage UML. Les sous-paragraphes suivants fournissent des spécifications plus détaillées de ces classes d'objets supports sous divers aspects.

2.2.1 Attributs et relations

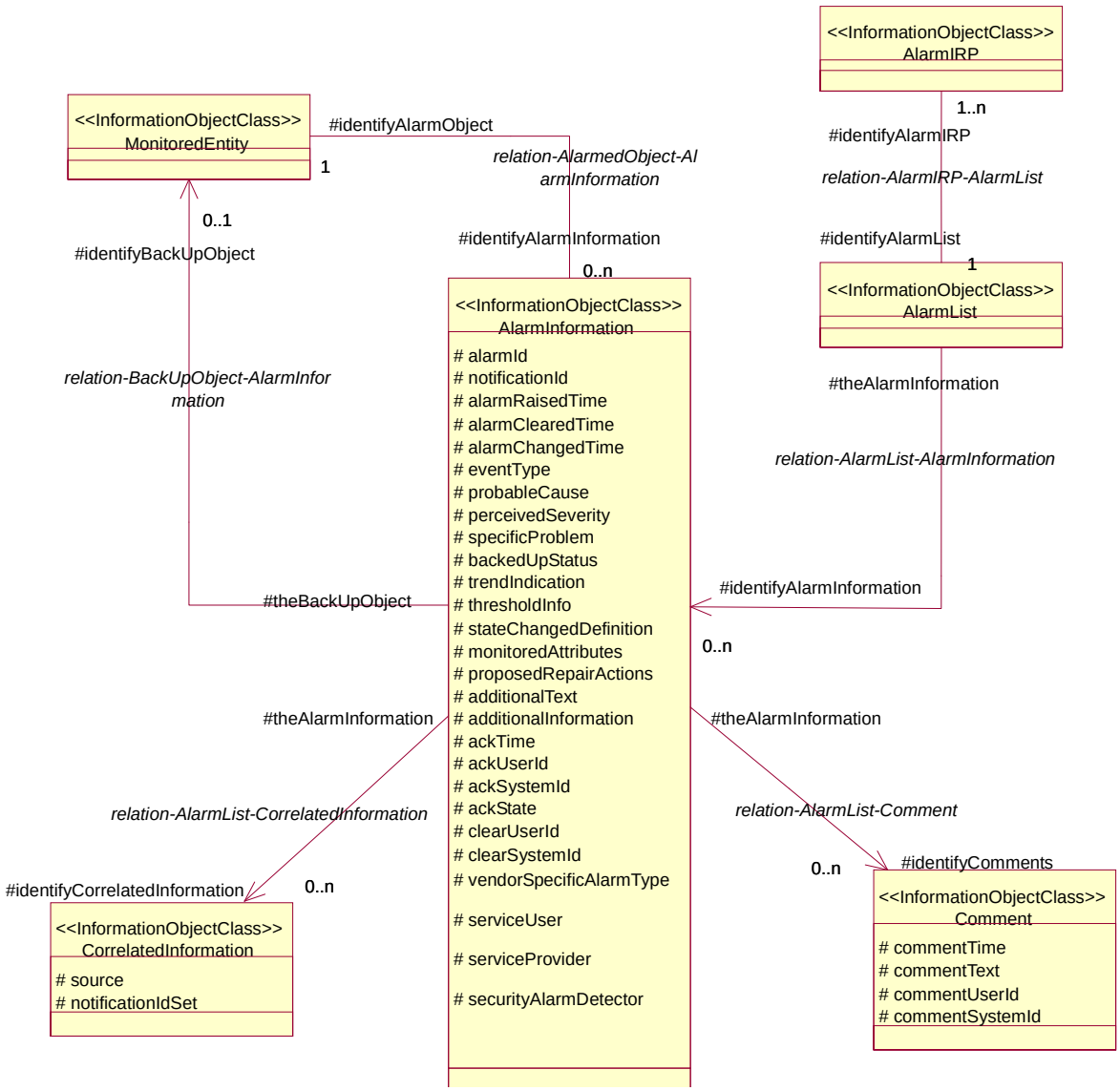


Figure II.1 – Classes d'objets informationnels de gestion des alarmes

2.2.2 Héritage

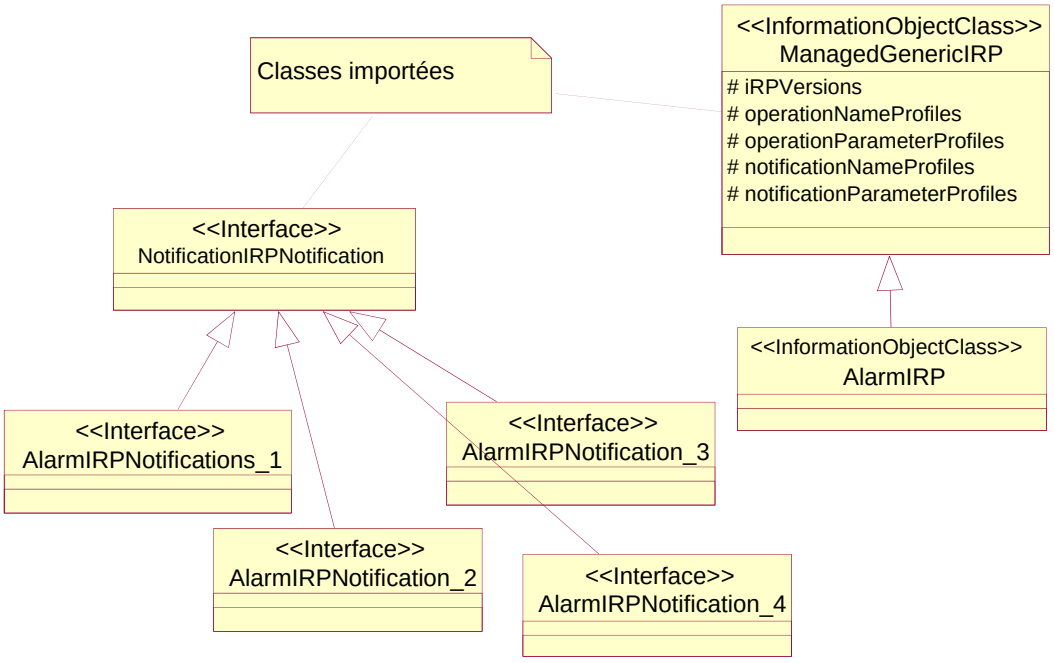


Figure II.2 – Héritage des classes IOC de gestion des alarmes

2.3 Définitions des classes d'objets informationnels

Nom de la classe	Qualificatif (M=obligatoire)	Identifiants des besoins
AlarmInformation	M	REQ-FM-FON-01, REQ-FM-FON-02, etc.
AlarmList	M	REQ-FM-FON-n
…		

2.3.1 AlarmInformation

2.3.1.1 Definition

AlarmInformation contient des informations relatives aux alarmes d'une entité gérée MonitoredEntity qui est en état d'alarme.

….

2.3.1.2 Attributs

Nom de l'attribut	Qualificatif de prise en charge (*Support qualifier*)	Qualificatif de lecture (*Read qualifier*)	Qualificatif d'écriture (*Write qualifier*)	Identifiants des besoins
alarmed	M	M	M	
probableCause	C	M	C	
structuredProbableCause	C	M	C	
perceivedSeverity	M	M	M	
specificProblem	O	O	O	
…				
…				

2.3.1.3 Diagramme d'états

Les alarmes possèdent des états.

…

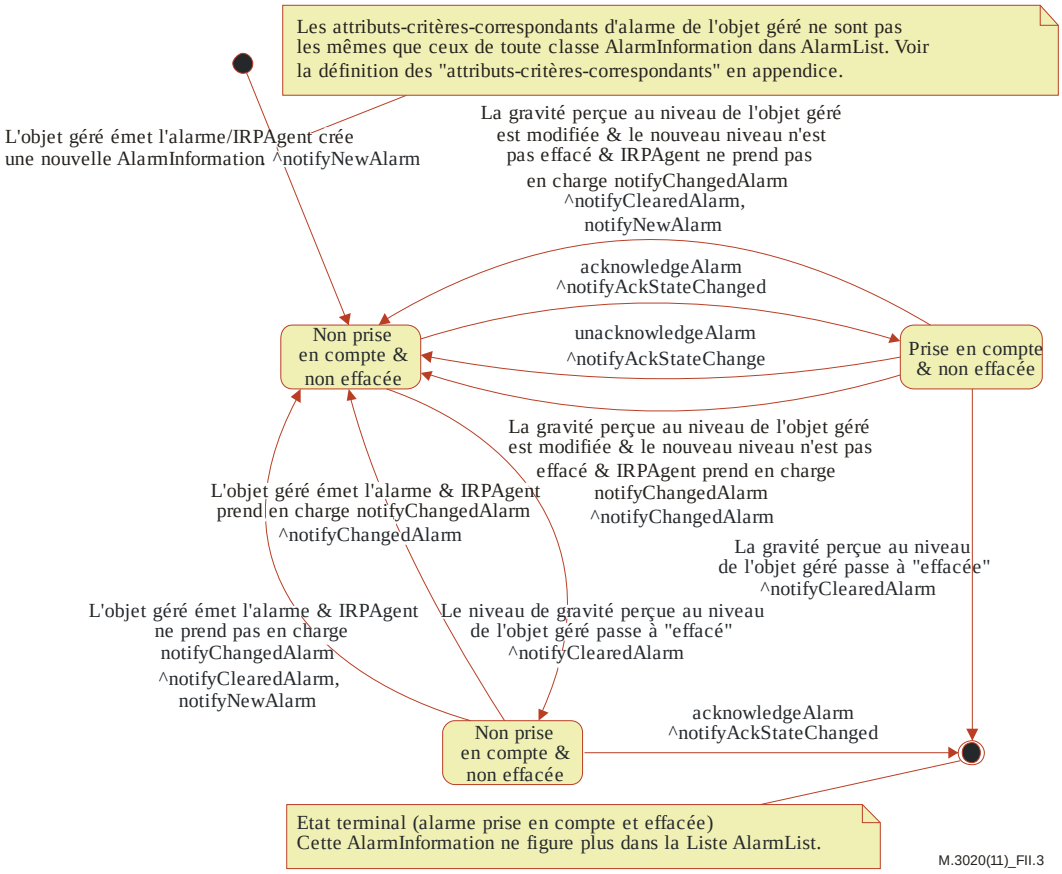


Figure II.3 – Diagramme d'états des informations d'alarme

2.3.2 AlarmList

2.4 Définitions des relations d'information

Relation	Qualificatif de prise en charge (*Support qualifier*)	Identifiants des besoins
relation-AlarmIRP-AlarmList	M	REQ-FM-FON-x
…		

2.4.1 relation-AlarmIRP-AlarmList (M)

2.4.1.1 Définition

Cette relation est la relation entre AlarmIRP et AlarmList.

2.4.1.2 Rôles

Nom	Définition
identifyAlarmIRP	Représente la possibilité d'obtenir les identités d'un ou plusieurs points AlarmIRP.
identifyAlarmList	Représente la possibilité d'obtenir l'identité d'une liste AlarmList.

2.4.1.3 Constrainte

Il n'y a pas de contrainte sur cette relation.

2.4.2 relation-AlarmList-AlarmInformation (M)

...

2.5 Définition des attributs d'information

2.5.1 Définition et valeurs autorisées

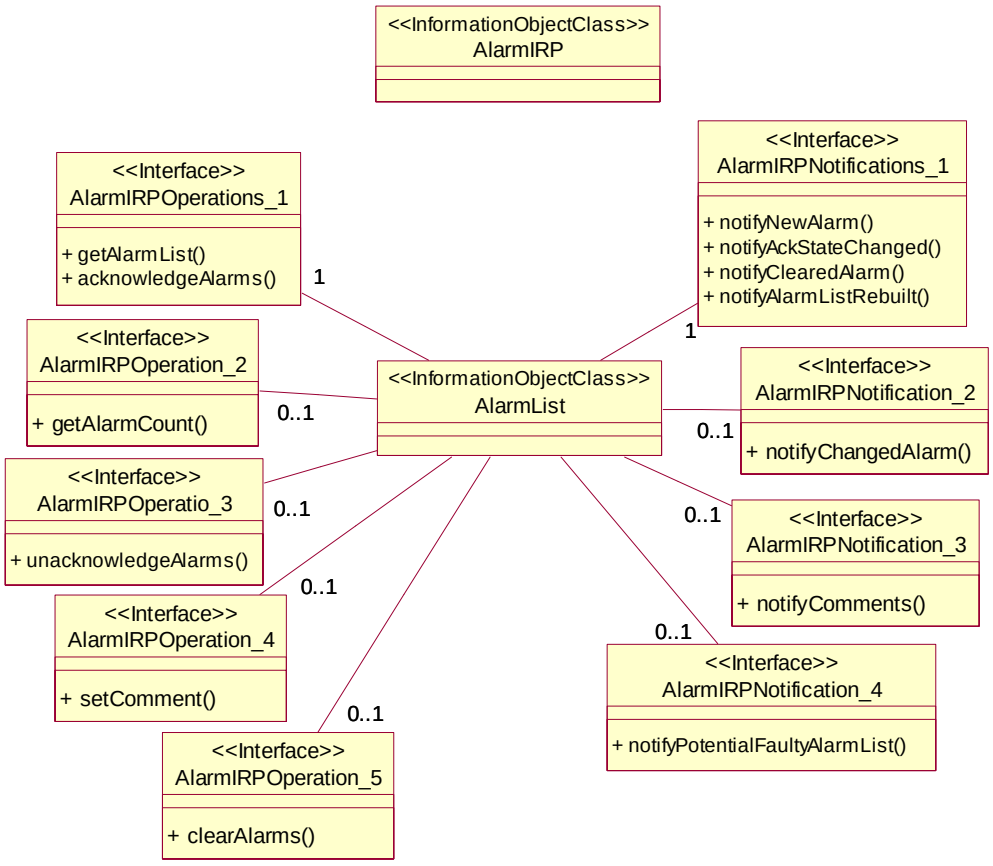
Nom	Définition	Type d'information/ Valeurs autorisées
en alarme	Identifie une AlarmInformation dans la liste AlarmList	INTEGER
notificationId	Identifie la notification qui achemine AlarmInformation.	INTEGER
ntfSusbcriptionState	Indique l'état d'activation d'un abonnement.	ENUMERATED/"suspendu": l'abonnement est suspendu. "nonSuspendu": l'abonnement est actif.

2.5.2 Constraintes

Nom	Attribut(s) concerné(s)	Définition
inv_notificationId	notificationId	Les NotificationId doivent être choisies de façon à être uniques parmi toutes les notifications d'un objet géré donné (représentant l'élément de réseau) tant que la corrélation des alarmes est pertinente. L'algorithme permettant d'accomplir la corrélation des alarmes sort du domaine d'application de cet IRP.

3 Définition d'interface

3.1 Diagrammes de classes représentant les interfaces



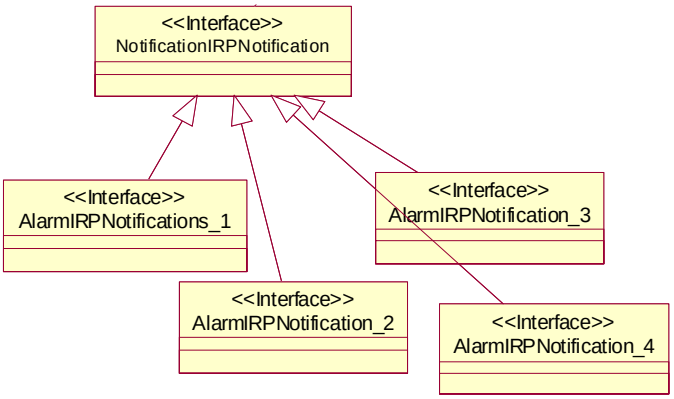


Figure II.4 – Diagramme de classes du point IRP de gestion des alarmes

3.2 Règles génériques

Règle 1: Chaque opération possédant au moins un paramètre d'entrée est assortie d'une précondition valid_input_parameter indiquant que tous les paramètres d'entrée doivent être valides au regard de leur type d'information. De plus, toutes les opérations de ce type sont assorties d'une exception operation_failed_invalid_input_parameter qui est levée lorsque la précondition valid_input_parameter est fausse. Les états d'entrée et de sortie de l'exception sont identiques.

Règle 2: Chaque opération possédant au moins un paramètre d'entrée optionnel est assortie d'un ensemble de préconditions supported_optional_input_parameter_xxx où "xxx" correspond au nom du paramètre d'entrée optionnel, la précondition indiquant que l'opération prend en charge le paramètre d'entrée optionnel désigné. De plus, toutes les opérations de ce type sont assorties d'une exception operation_failed_unsupported_optional_input_parameter_xxx qui est levée lorsque:

a) la précondition supported_optional_input_parameter_xxx est fausse; et

b) le paramètre d'entrée optionnel désigné achemine de l'information.

Les états d'entrée et de sortie de l'exception sont identiques.

Règle 3: Chaque opération doit prendre en charge une exception générique operation_failed_internal_problem, qui est levée lorsqu'un problème interne survient et que l'opération ne peut pas être menée à terme. Les états d'entrée et de sortie de l'exception sont identiques.

3.3 Interface AlarmIRPOperations_1 (O)

Nom de l'opération	Qualificatif (M=Obligatoire)	Identifiants des besoins
acknowledgeAlarms	M	REQ-FM-FON-x, REQ-FM-FON-y
getAlarmList	M	…

3.3.1 Operation acknowledgeAlarms (M)

3.3.1.1 Définition

Le Gestionnaire invoque cette opération pour accuser réception d'une ou plusieurs alarmes.

3.3.1.2 Paramètres d'entrée

Nom du paramètre	Qualificatif de prise en charge (*Support qualifier*)	Type d'information/ Valeurs autorisées	Remarque
…			
eventIdList	M	SET OF INTEGER/–	Liste des alarmes dont il faut accuser réception.

3.3.1.3 Paramètres de sortie

Nom du paramètre	Qualificatif de prise en charge (*Support qualifier*)	Information de correspondance/ Type d'information/ Valeurs autorisées	Remarque
…			
Status	M	-- / ENUM / "OperationSucceeded": si allAlarmsAcknowledged est vrai, "OperationPartiallySucceeded": si someAlarmAcknowledged est vrai, "OperationFailed": si operationFailed est vrai.	

3.3.1.4 Précondition

atLeastOneValidId.

Nom de l'affirmation	Définition
atLeastOneValidId	La liste AlarmInformationReferenceList contient au moins un identifiant qui identifie une AlarmInformation dans la liste AlarmList, et cette AlarmInformation identifiée doit avoir son état ackState positionné à "unacknowledged" et le même attribut perceivedSeverity (si cet attribut est fourni).

3.3.1.5 Postcondition

someAlarmAcknowledged OU allAlarmsAcknowledged.

Nom de l'affirmation	Définition
someAlarmAcknowledged	…
allAlarmsAcknowledged	…

3.3.1.6 Exceptions

Nom	Définition
operation_failed	**Condition:** La précondition est fausse ou la postcondition est fausse. **Information renvoyée:** Le statut du paramètre de sortie. **Etat de sortie:** Etat d'entrée.

3.3.2 Opération getAlarmList (M)

…

Appendice III

Comparaison avec la Recommandation UIT-T Z.601

(Cet appendice ne fait pas partie intégrante de la présente Recommandation.)

Le présent appendice fournit des informations sur la relation entre la présente Recommandation et [b-UIT-T Z.601], qui est utilisée pour l'élaboration des Recommandations de la série UIT-T M.1400.

La présente Recommandation décrit une méthodologie pour la spécification d'interfaces de gestion entre deux systèmes physiques, alors que [b-UIT-T Z.601] fournit un cadre pour le développement d'un seul système. Cette dernière est une architecture de données qui recense, au sein d'un seul et même système, des entités susceptibles d'être retenues comme interfaces, ainsi que les interfaces situées à la frontière de ce système, lesquelles seront des interfaces entre systèmes.

La méthodologie décrite dans la présente Recommandation est avant tout destinée au développement d'un ensemble de Recommandations d'interfaces de gestion plutôt que de systèmes individuels. L'architecture des données décrite dans [b-UIT-T Z.601] n'impose pas de procéder à une expression des besoins comme c'est le cas pour la présente Recommandation (phase d'expression des besoins), étant donné qu'elle prescrit uniquement la spécification de systèmes pris isolément et non leur finalité vis-à-vis d'une organisation.

La Recommandation [b-UIT-T Z.601] met l'accent sur la spécification de la terminologie et de la grammaire externes telles qu'elles sont perçues par les utilisateurs finals. La présente Recommandation s'intéresse à la spécification d'interfaces de gestion, qui ne sont pas nécessairement vues des utilisateurs finals.

Dans la présente Recommandation, les besoins correspondant au problème à résoudre se répartissent en deux catégories. La première est dénommée "besoins de niveau métier", la seconde "besoins de niveau spécification". Les besoins de niveau spécification peuvent éventuellement inclure des besoins concernant l'interaction avec l'utilisateur final, et ce au niveau de leurs interfaces homme-machine. Certains de ces besoins peuvent spécifier des exigences syntaxiques qui devront être mises en œuvre via toute interface de gestion. Dans [b-UIT-T Z.601], les exigences syntaxiques correspondent aux schémas de terminologie externe de l'architecture des données.

Le résultat de la phase d'analyse est un modèle d'information. Dans [b-UIT-T Z.601], ce modèle d'information correspond à un schéma de concept de l'architecture des données. Si les modèles d'information issus de la phase d'analyse n'expriment pas toutes les informations nécessaires tirées des exigences syntaxiques, il faudra éventuellement intégrer, dans la conception d'implémentation, un mappage de ces besoins.

La documentation de la phase de conception d'implémentation sera constituée de deux parties:

1) Une spécification de données dépendante de la technologie commune à plusieurs interfaces (au moyen par exemple de GDMO ou de CORBA IDL), qui correspond à un schéma de terminologie interne au sens de l'architecture des données décrite dans [b-UIT-T Z.601].
2) Une spécification de chaque interface, dépendante de la technologie (au moyen par exemple de CMIP ou de CORBA IDL), qui correspond à un schéma de diffusion (*distribution schema*) au sens de l'architecture des données décrite dans [b-UIT-T Z.601].

Appendice IV

Questions appelant un complément d'étude

(Cet appendice ne fait pas partie intégrante de la présente Recommandation.)

Le présent appendice dresse une liste des questions connues qui appellent un complément d'étude.

IV.1 SOA

L'approbation de la Recommandation [UIT-T M.3060] (*Principes pour la gestion des réseaux de prochaine génération*) a marqué un tournant dans la gestion des réseaux, qui est passée d'une approche orientée objet à une approche orientée vers les services. Il sera nécessaire d'examiner les répercussions de ce changement afin de recenser les éventuelles modifications à apporter aux futures révisions de la présente Recommandation.

IV.2 UML

La présente version de la Recommandation UIT-T M.3020 fait référence à la version 1.5 du langage UML par souci de cohérence avec les spécifications correspondantes du 3GPP. Une version révisée de la Recommandation UIT-T M.3020 devra faire référence aux futures versions d'UML:

– Le méta-métamodèle MOF de l'OMG intègre UML 2.x sous forme de métamodèle pris en charge par les grands fabricants d'outils du secteur. Avant la version 2.0 d'UML, il n'y avait pas de méta-métamodèle général et le langage UML lui-même n'était pas une norme. MOF prend en charge l'ajout et la création de nouveaux métamodèles définis de façon précise via OCL, langage de calcul basé sur la logique des prédicats.

– Tant les secteurs concernés (télécommunications, pouvoirs publics et armée) que les fabricants d'outils convergent vers le modèle MOF de l'OMG.

– Le méta-métamodèle MOF présente plusieurs avantages: il prend en charge une famille de métamodèles qui peuvent servir à définir des modèles d'objets, des relations IHM et diverses implémentations dépendantes d'une technologie; de plus, il permet d'opérer des transformations entre modèles de façon normalisée. UML 1.5 n'offre pas ces possibilités, car cette version du langage existe indépendamment d'un métamodèle de niveau supérieur.

IV.3 Visibilité

Il a été proposé que la visibilité par défaut soit "privée" pour les attributs et "publique" pour les opérations de façon à encourager l'encapsulation des données et à limiter le temps et le travail consacrés à la définition du modèle d'implémentation.

IV.4 Définitions des types

Lors de l'élaboration d'une nouvelle spécification basée sur la méthodologie décrite dans le présent document, il est nécessaire de préciser les types des paramètres et des attributs. La version actuelle de la présente Recommandation ne donnant pas de définitions de types formelles, les définitions de types ayant la même signification risquent d'être différentes et incohérentes selon les spécifications. Par exemple, un tableau de nombres entiers pourra être défini comme une liste de nombres entiers, une séquence de nombres entiers ou un ensemble de nombres entiers.

L'Annexe E dresse la liste des types pouvant être utilisés dans le modèle conceptuel.

Appendice V

Exemples supplémentaires d'utilisation du langage UML

(Cet appendice ne fait pas partie intégrante de la présente Recommandation.)

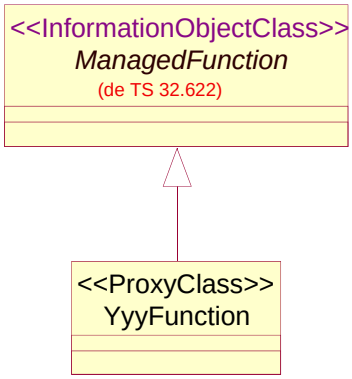
Le présent appendice contient des exemples supplémentaires de l'utilisation du langage UML telle que décrite à l'Annexe C.

V.1 Classe Proxy

V.1.1 Premier exemple

Cet exemple illustre une <<ProxyClass>> appelée YyyFunction. Il représente toutes les classes IOC énumérées dans la Note figurant dans le diagramme UML. Dans le cadre de cet exemple, toutes les IOC énumérées héritent de la classe IOC ManagedFunction.

Grâce à <<ProxyClass>>, il n'est plus nécessaire de représenter de nombreuses boîtes UML <<InformationObjectClass>>, en l'occurrence celles dont le nom est indiqué dans la Note, sur le diagramme UML.



NOTE – La <<ProxyClass>> YyyFunction représente AsFunction, AucFunction, BgFunction, etc.

Notation <<ProxyClass>> – Exemple V.1

V.1.2 Deuxième exemple

Cet exemple illustre une <<ProxyClass>> appelée YyyFunction. Il représente toutes les classes IOC énumérées dans la Note figurant juste au-dessous du diagramme UML. Dans le cadre de cet exemple, toutes les IOC énumérées ont des relations (internes et externes) de type lien.

Les véritables noms de la classe IOC représentée par la <<ProxyClass>> InternalYyyFunction et par la <<ProxyClass>> ExternalYyyFunction sont énumérés au sous-paragraphe X.Y de la <<ProxyClass>> YyyFunction associée. Par exemple, sous le paragraphe X.Y.1 correspondant à la classe AsFunction, deux paragraphes sont ajoutés pour énumérer toutes les entités internes homologues et toutes les entités externes qui sont liées à AsFunction. Voir ci-dessous le texte entre guillemets dans lequel AsFunction est pris comme exemple de YyyFunction.

Les véritables noms de la classe IOC représentée par la <<ProxyClass>> Link_a_z et par la <<ProxyClass>> ExternalLink_a_z sont énumérés au sous-paragraphe de X.Y de la <<ProxyClass>> YyyFunction associée. Par exemple, sous X.Y.1 correspondant à AsFunction, deux paragraphes sont ajoutés pour énumérer les noms des IOC représentées par Link_a_z et par ExternalLink_a_z. Voir ci-dessous le texte entre guillemets dans lequel AsFunction est pris comme exemple de YyyFunction.

"

X.Y.1 AsFunction

X.Y.1.1 Définition

La classe IOC représente la fonctionnalité As. Pour de plus amples informations sur As, voir [b-3GPP TS 23.002].

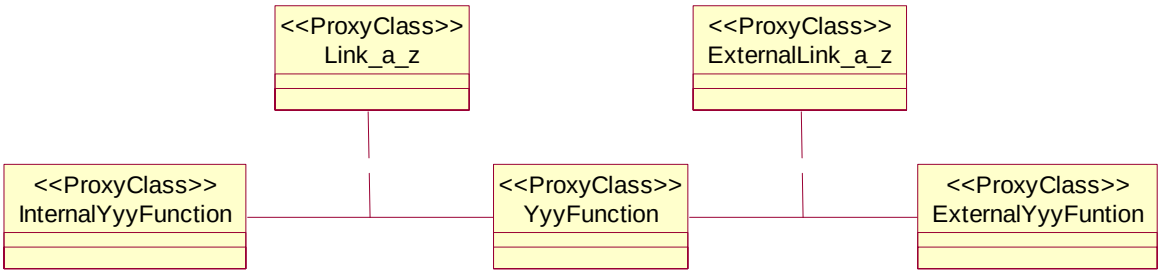
Les <<ProxyClass>> InternalYyyFunction liées représentent SlsFunction, CscfFunction, HlrFunction...

La <<ProxyClass>> ExternalYyyFunction liée représente...

La <<ProxyClass>> Link_a_z représente Link_As_Scscf, Link_Bgcf_Scscf …

La <<ProxyClass>> ExternalLink_a_z représente...

"



NOTE – La partie 'Yyy' de la <<ProxyClass>> YyyFunction représente AsFunction, AucFunction, BgFunction, etc.

Notation <<ProxyClass>> – Exemple V.2

Appendice VI

Lignes directrices pour la numérotation des besoins

(Cet appendice ne fait pas partie intégrante de la présente Recommandation.)

Le format utilisé pour numéroter un besoin est le suivant:

REQ-Etiquette-Catégorie-Numéro

où "Etiquette" est une abréviation correspondant à la Recommandation (ou à une partie de celle-ci). L'ensemble des étiquettes n'est ni limité ni normalisé. L'ensemble des catégories est défini dans la présente Recommandation.

Quelques questions:

– Comment structurer l'étiquette dans une spécification de besoins de grande taille?

– Comment gérer la suppression et l'ajout de besoins?

On pourra s'aider des lignes directrices suivantes:

– Les besoins ne devraient jamais être renumérotés. On peut déroger à cette règle lors de la première publication d'une spécification et seulement dans ce cas, mais il peut être préférable, même dans ce cas, d'éviter la renumérotation, car la spécification peut avoir été utilisée aussi dans sa version provisoire.

– Etant donné que les besoins ne doivent pas être renumérotés, on ne peut pas partir du principe qu'ils seront numérotés séquentiellement sur l'ensemble de la spécification.

– L'étiquette peut servir à diviser la numérotation en parties logiques. Par exemple, le format "A_B" est recommandé pour indiquer que "B" est une partie logique de "A". On pourra toutefois adopter d'autres formats tant que la structure basée sur le symbole "-" servant à séparer les champs du numéro de besoin est conservée.

– Il n'est pas recommandé d'utiliser une notation avec préfixe ou suffixe (c'est-à-dire l'ajout de caractères avant ou après "Numéro"), car la partie "Numéro" n'est pas censée véhiculer une information sémantique.

– Comme solution de substitution au format "A_B", les auteurs de la spécification pourront décider d'allouer un intervalle de numéros à chaque groupe de besoins. Cette solution devrait être autorisée.

Références bibliographiques

[b-UIT-T M.1401] Recommandation UIT-T M.1401 (2006), *Formalisation des désignations d'interconnexion entre réseaux de télécommunication des opérateurs.*

[b-UIT-T M.1403] Recommandation UIT-T M.1403 (2007), *Formalisation des commandes génériques.*

[b-UIT-T M.1404] Recommandation UIT-T M.1404 (2007), *Formalisation des commandes relatives aux interconnexions entre réseaux d'opérateurs*.

[b-UIT-T Z.601] Recommandation UIT-T Z.601 (2007), *Architecture des données d'un système logiciel*.

[b-3GPP TS 23.002] 3GPP TS 23.002 (en vigueur), *Network architecture*.

[b-3GPP TS 32.101] 3GPP TS 32.101 V10.0.0 (2010), *Telecommunication management; Principles and high level requirements*.

[b-3GPP TS 32.150] 3GPP TS 32.150 V10.2.0 (2011), *Telecommunication management; Integration Reference Point (IRP) Concept and definitions.*

[b-3GPP TS 32.151] 3GPP TS 32.151 V10.1.0 (2010), *Telecommunication management; Integration Reference Point (IRP) Information Service (IS) template*.

[b-3GPP TS 32.152] 3GPP TS 32.152 V10.0.0 (2010), *Telecommunication management; Integration Reference Point (IRP) Information Service (IS) Unified Modelling Language (UML) repertoire*.

[b-3GPP TS 32.302] 3GPP TS 32.302 V10.0.0 (2010), *Telecommunication management; Configuration Management (CM); Notification Integration Reference Point (IRP); Information Service (IS).*

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	Systèmes audiovisuels et multimédias
Série I	Réseau numérique à intégration de services
Série J	Réseaux câblés et transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	**Gestion des télécommunications y compris le RGT et maintenance des réseaux**
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Terminaux et méthodes d'évaluation subjectives et objectives
Série Q	Commutation et signalisation
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	Terminaux des services télématiques
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	Réseaux de données, communication entre systèmes ouverts et sécurité
Série Y	Infrastructure mondiale de l'information, protocole Internet et réseaux de prochaine génération
Série Z	Langages et aspects généraux logiciels des systèmes de télécommunication

Imprimé en Suisse
Genève, 2014